Google TPU2 第一款训练卡

ZOMI

Talk Overview

1. **AI 计算体系**
   - 深度学习计算模式
   - 计算体系与矩阵运算
2. **AI 芯片基础**
   - 通用处理器 CPU
   - 通用图形处理器 GPU
   - AI专用处理器 NPU/TPU
3. **GPU详解**
   - 英伟达GPU架构发展
   - Tensor Core和NVLink
4. **国外 AI 芯片**
   - 特斯拉 DOJO 系列
   - 谷歌 TPU 系列
5. **国内 AI 芯片**
   - 壁仞科技芯片架构
   - 寒武纪科技芯片架构
6. **AI芯片的思考**
   - SIMD&SIMT与编程体系
   - AI芯片的架构思路与思考

Talk Overview

I. 国外 AI 芯片

- 英伟达 GPU 芯片架构剖析
- 特斯拉 DOJO 芯片架构剖析
- 谷歌 TPU 芯片架构剖析

Talk Overview

I. 国外 AI 芯片

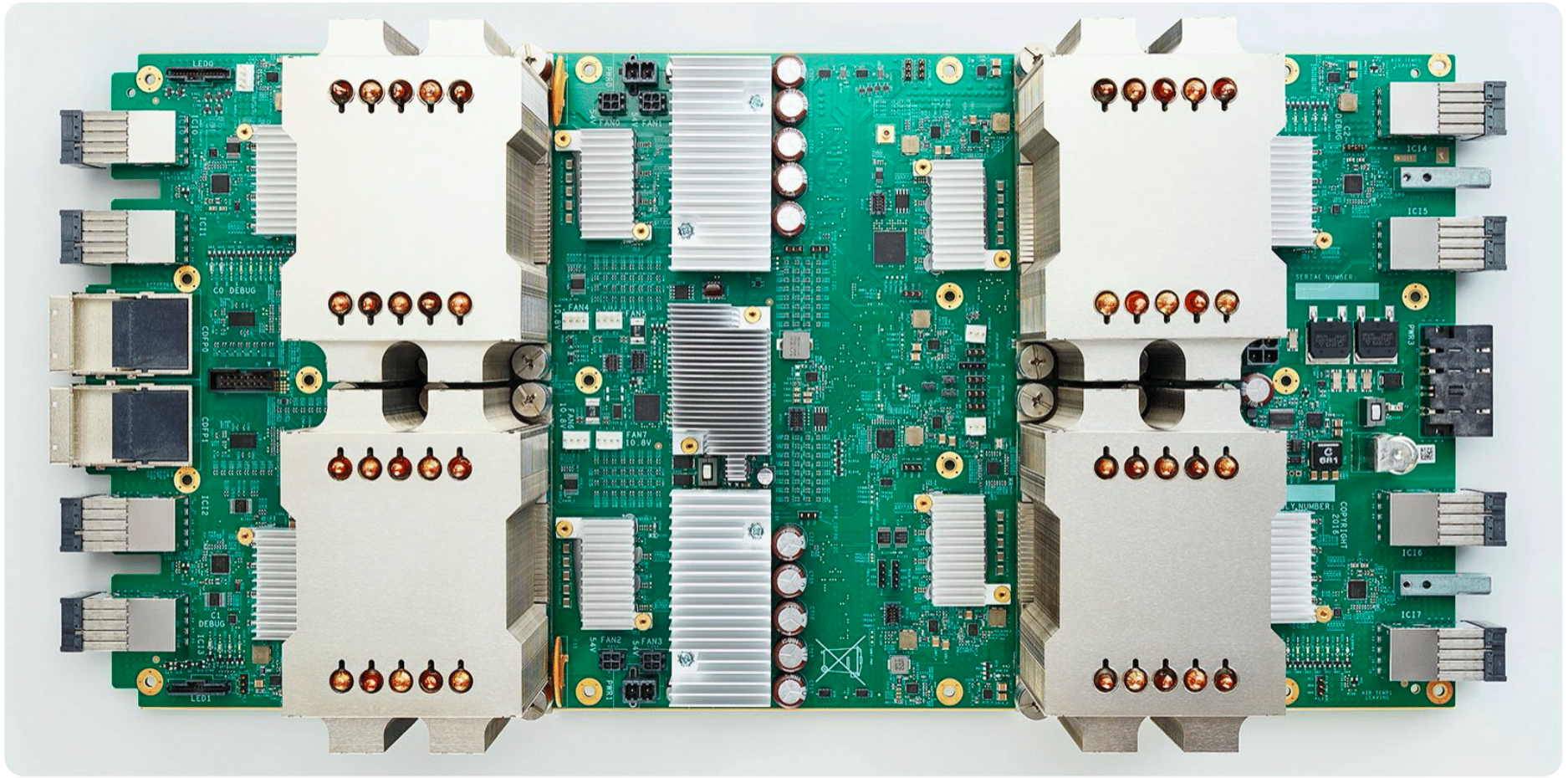
- 英伟达 GPU 芯片架构剖析
- 特斯拉 DOJO 芯片架构剖析
- 谷歌 TPU 芯片架构剖析

- TPU 历史发展
- TPU1 脉动阵列细节
- TPU2 第一款训练卡
- TPU3 性能 POD 超算
- TPU4 超级互联

why so many TPU ?

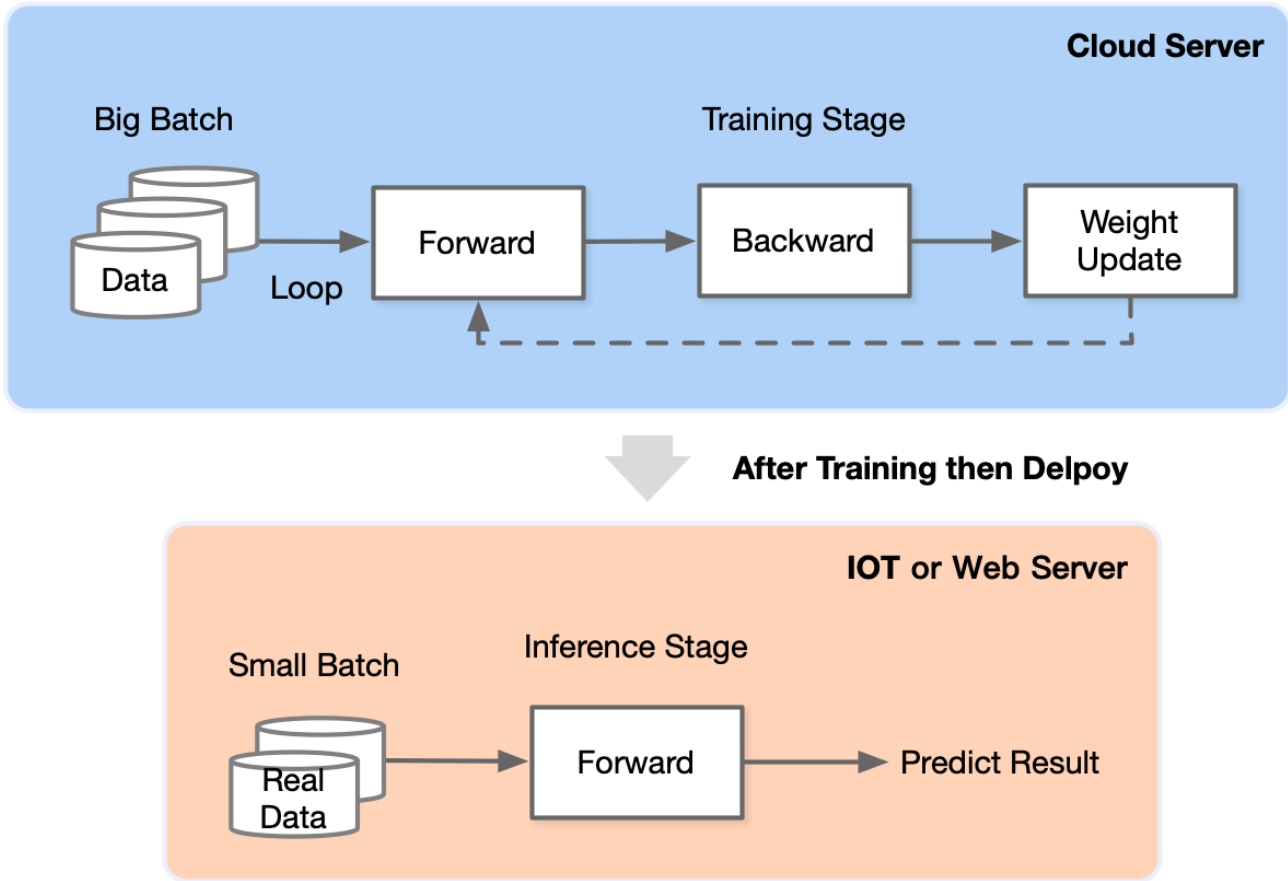
- why so many TPU ?

TPU2 产品形态



业务场景变化：从TPU V1 推理到 TPU V2 训练

- **训练过程**通过设计合适 AI 模型结构以及损失函数和优化算法，将数据集以 mini-batch 反复进行前向计算并计算损失，反向计算梯度利用优化函数来更新模型，使得损失函数最小。训练过程最重要是梯度计算和反向传播。
- **推理**在训练好的模型结构和参数基础上，一次前向传播得到模型输出过程。相对于训练，推理不涉及梯度和损失优化。最终目标是将训练好的模型部署生产环境中。



训练的难点

1. **训练并行化更难**：推理阶段，每个推理任务都是独立的，因此 DSA 芯片集群可以横向扩展。训练一个简单的小模型就需要迭代运行百万次，需要协调跨集群并行资源；
2. **更复杂的计算**：反向传播（自动微分）需要计算模型每一阶段权重参数和输入数据的梯度（导数），包括数据格式更高精度的激活值和转置权重矩阵 W^T W^TW 的乘法 😖；
3. **需要更大内存**：权重更新数据来自前向和反向传播的临时变量，临时变量需要被保留，因而提高了显存要求；在大模型临时变量一般为权重的2~4倍；

训练的难点

1. **更具可编程性：**训练算法和模型快速变化，设计期间仅限于当前最佳实践算法的 DSA 可能会很快过时，XXX 😔。
2. **高精度数据格式：**整数 INT8 可以用于推理，但是训练期间要充分捕捉梯度信息，通常需要FP16、 BF16和FP32等混合精度运算。

- 只要将 TPUv1 架构设计稍作更改，就可以得到 TPUv2 😄
- 而对 TPUv2 架构的修改，就是训练与推理的差别 😍

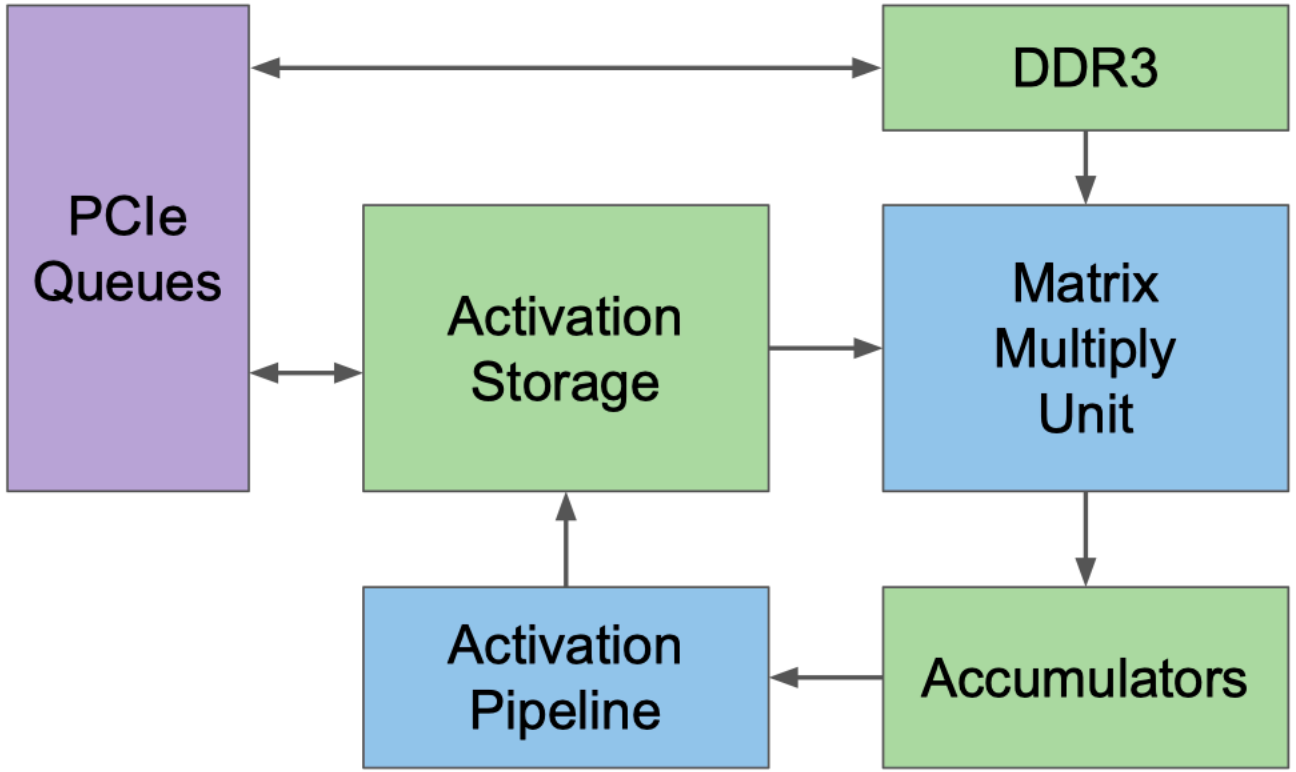
TPU v2 设计遵循 TPU v1👼：

使用大型二维矩阵乘法单元（Matrix Multiply Unit）脉动阵列来减少面积和能耗，以及增加一个可编程的片上存储，而不是 Cache。

1. TPU2 vs TPU1

TPUv2 Changes

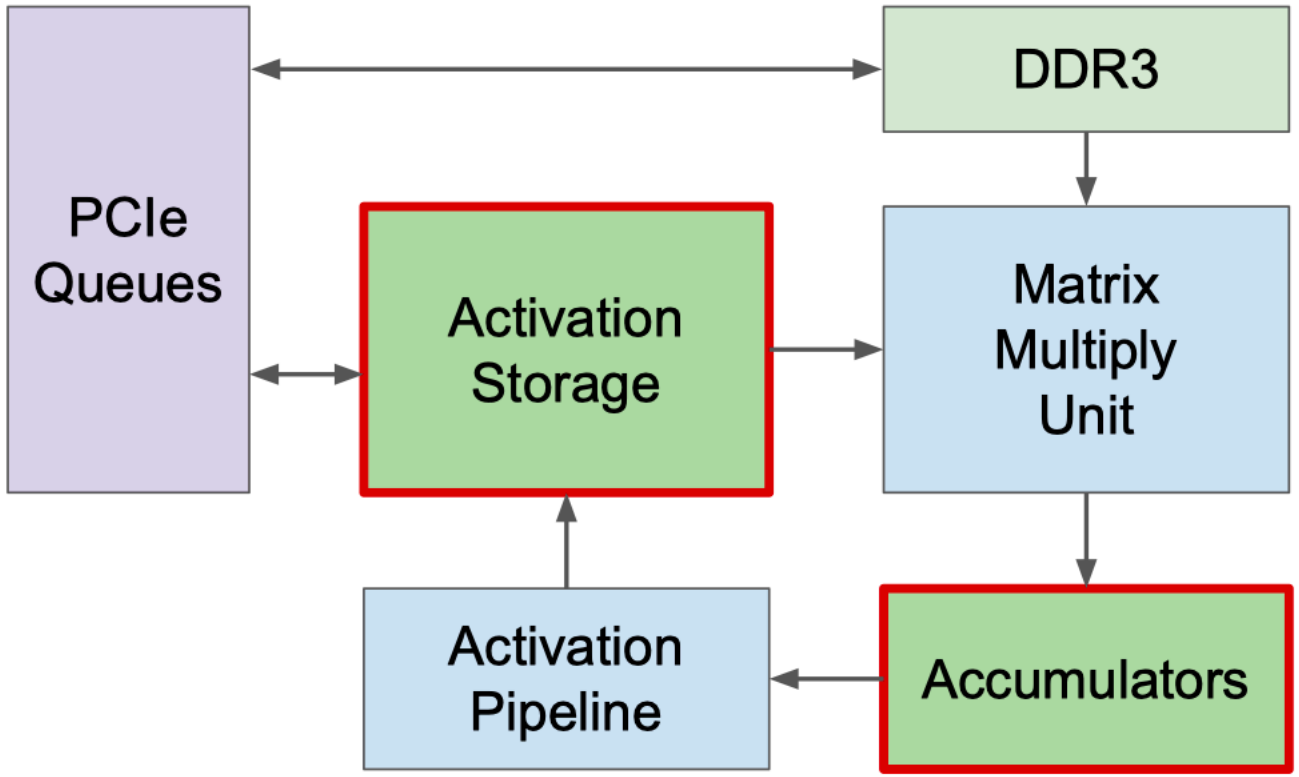
- TPU v1 只支持 INT8 计算，对训练而言动态范围不够大，因此 Google 在 TPU v2 引入 BFloat16 用于训练;
- 训练并行比推理的并行更难。由于针对的是训练而非推理，所以 TPU v2 可编程性也比 TPU v1 更高。



TPUv2 Changes 1

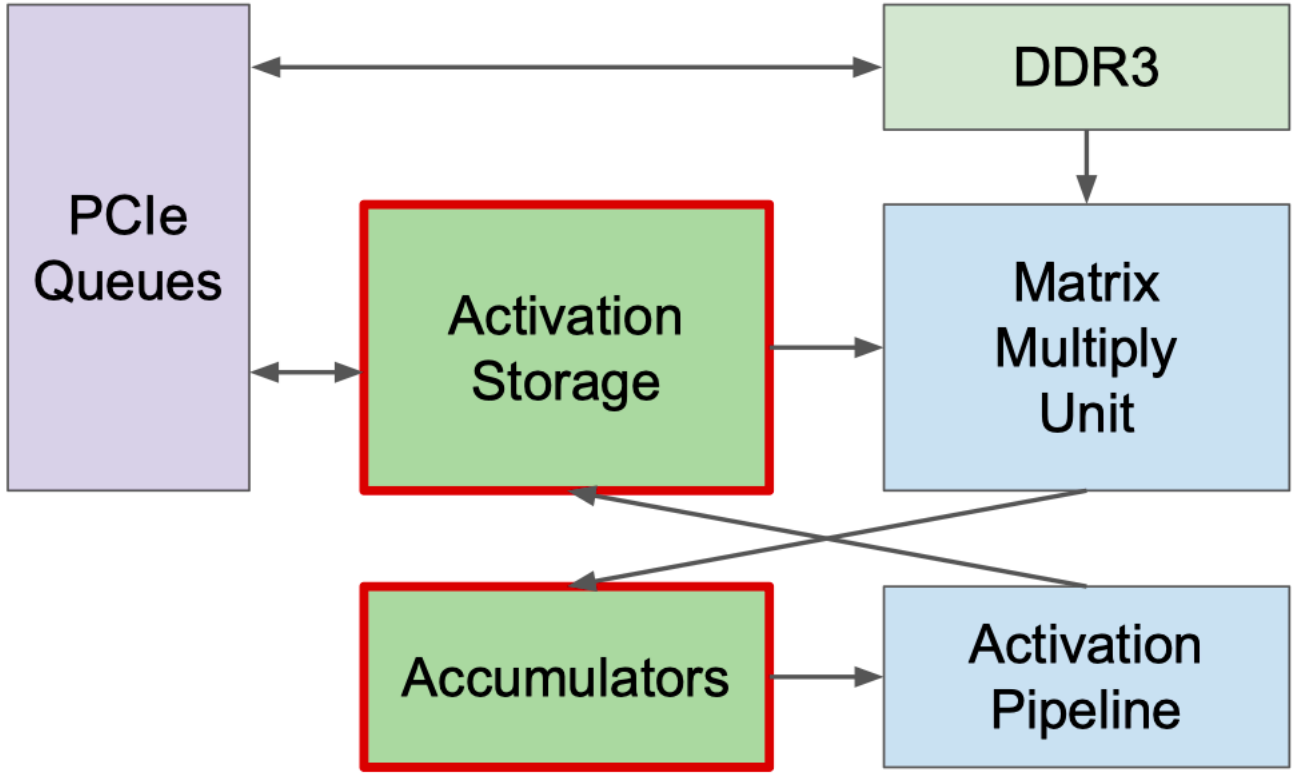
- TPU v1有两个存储区域：Accumulator 和 Activation Storage：

1. Accumulator 负责储存矩阵相乘结果；
2. Activation Storage 负责储存激活函数输出。



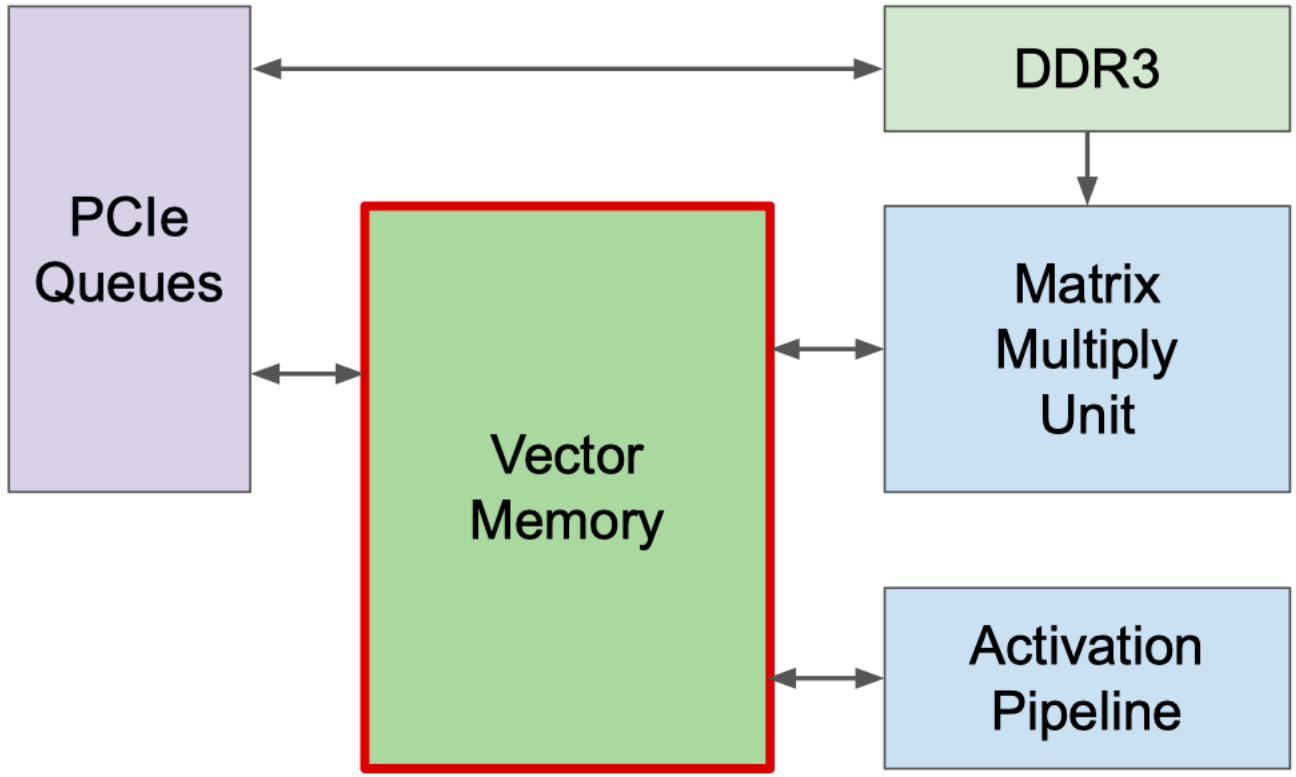
TPUv2 Changes 1

- TPU v2 将 Accumulator 和 Activation Storage 两个互相独立的缓冲区调整位置。
- 合并为向量存储区（Vector Memory），从而提高可编程性，这也更类似传统的L1 Cache。



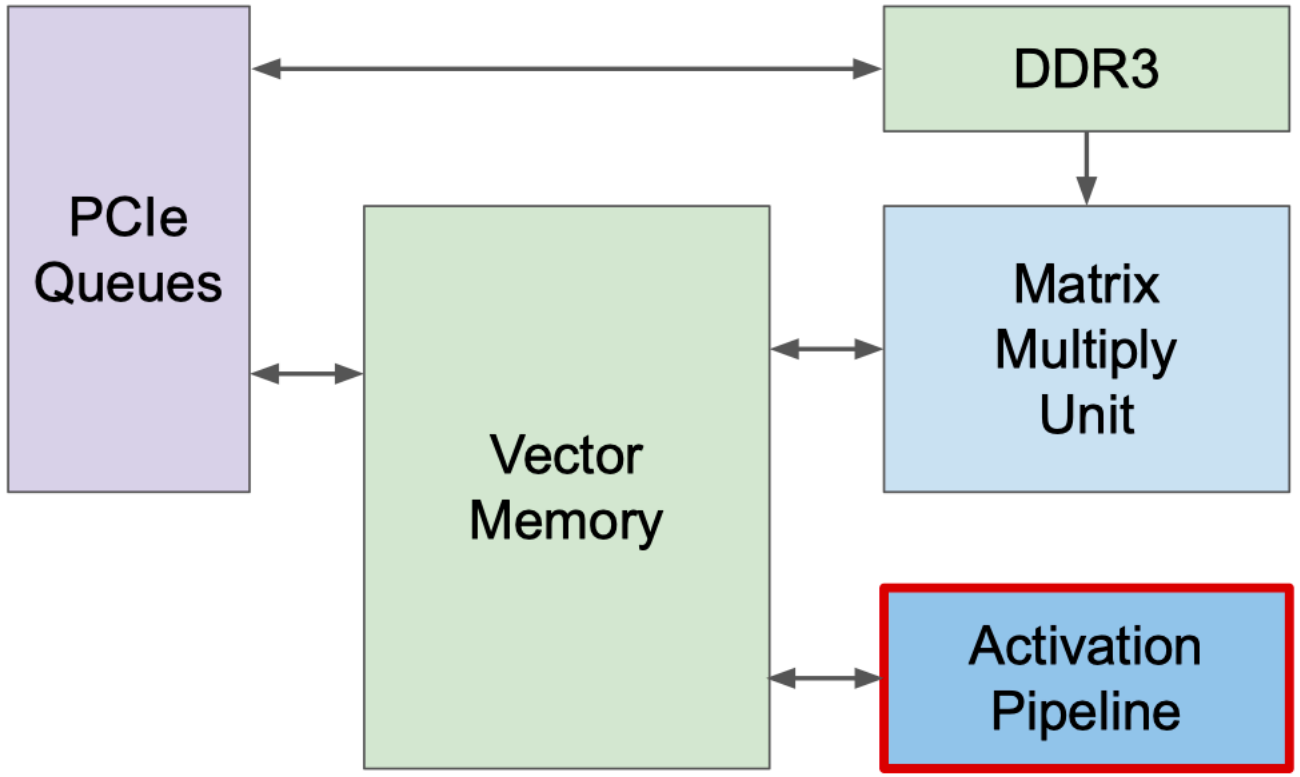
TPUv2 Changes 1

- TPU v2 将 Accumulator 和 Activation Storage 两个互相独立的缓冲区调整位置。
- 合并为向量存储区（Vector Memory），从而提高可编程性，这也更类似传统的L1 Cache。



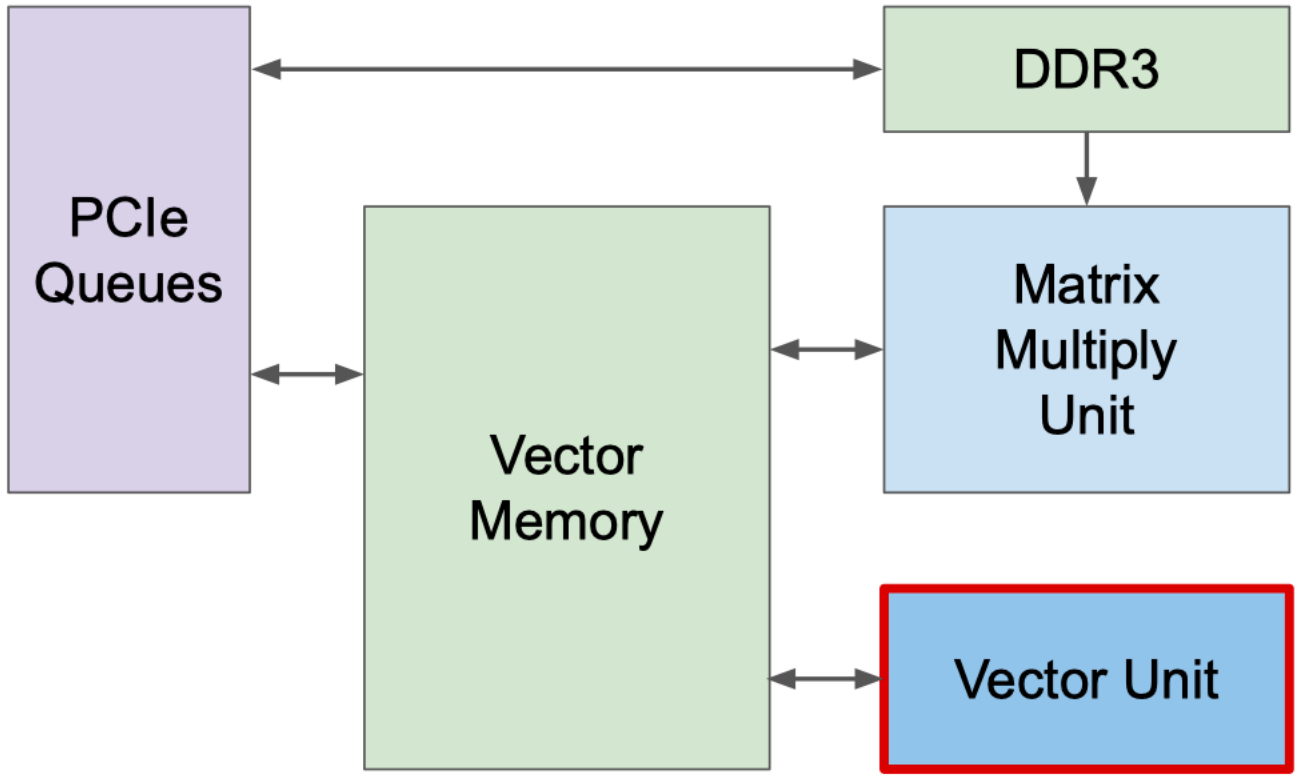
TPUv2 Changes 2

- 单个 Vector 存储器，而不是固定功能单元之间的缓冲区。



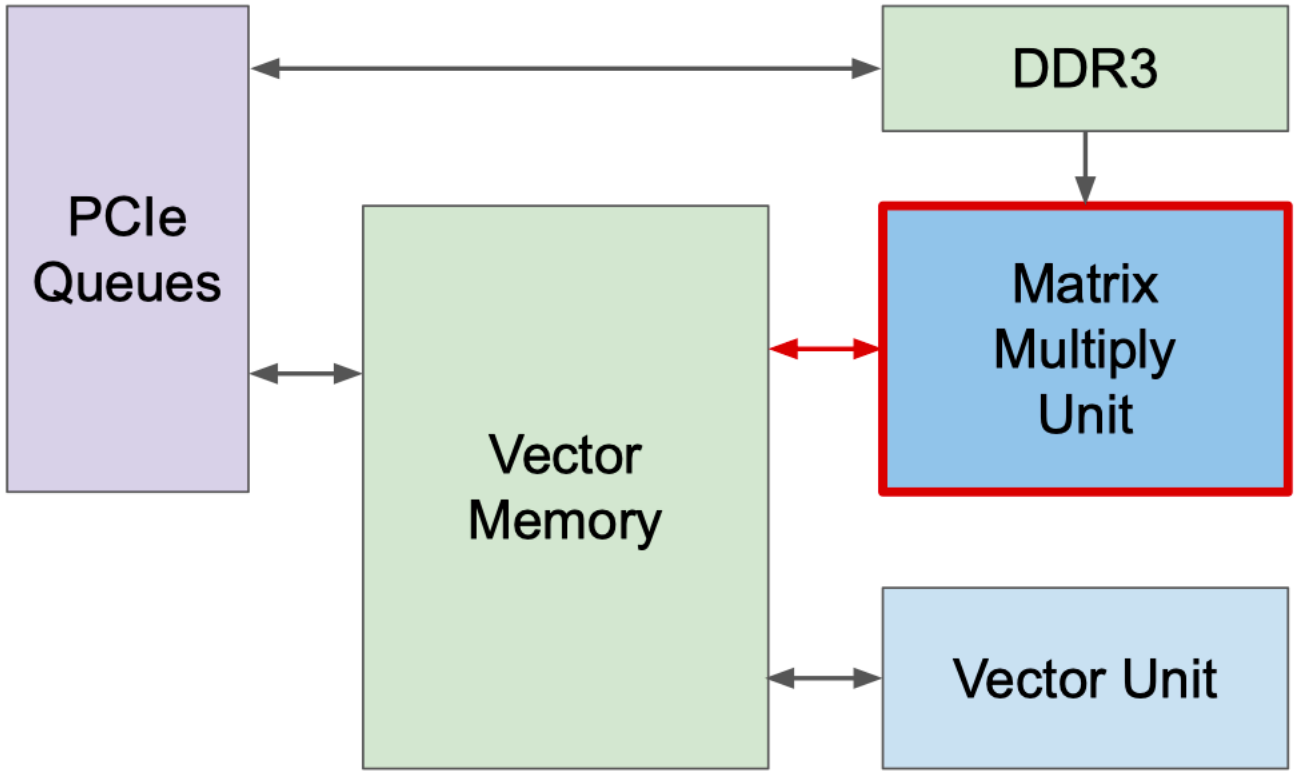
TPUv2 Changes 2

- 针对激活函数管道（Activation Pipeline），TPU v1 Pipeline 包含一组负责非线性激活函数运算的固定功能单元。
- TPU v2 则将其改为可编程的向量单元（Vector Unit），使其对编译器和编程人员而言更易用。



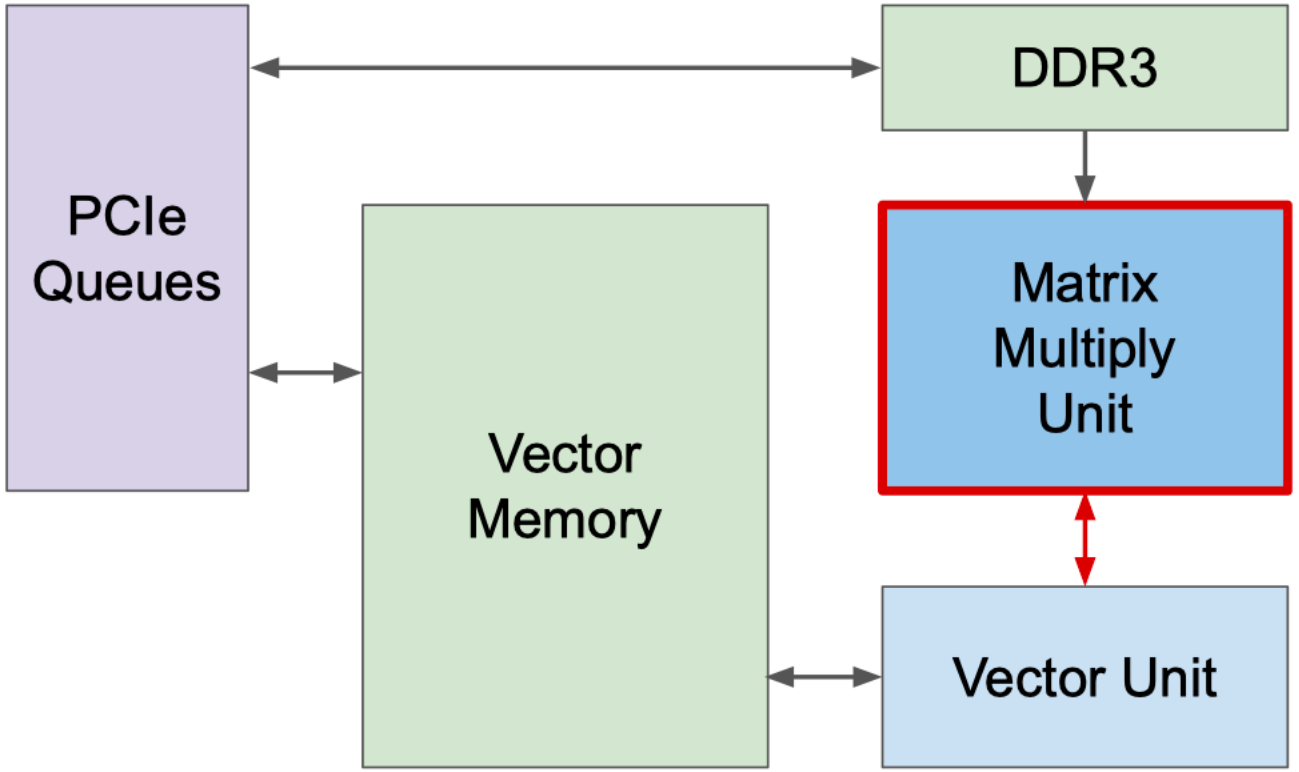
TPUv2 Changes 3

- 将矩阵乘法单元 MXU 直接与向量存储区 VU 连接。
- 如此一来，矩阵乘法单元 MXU 就成为向量单元 VU 的协处理器。
- 这种结构对编译器和编程人员而言更友好。



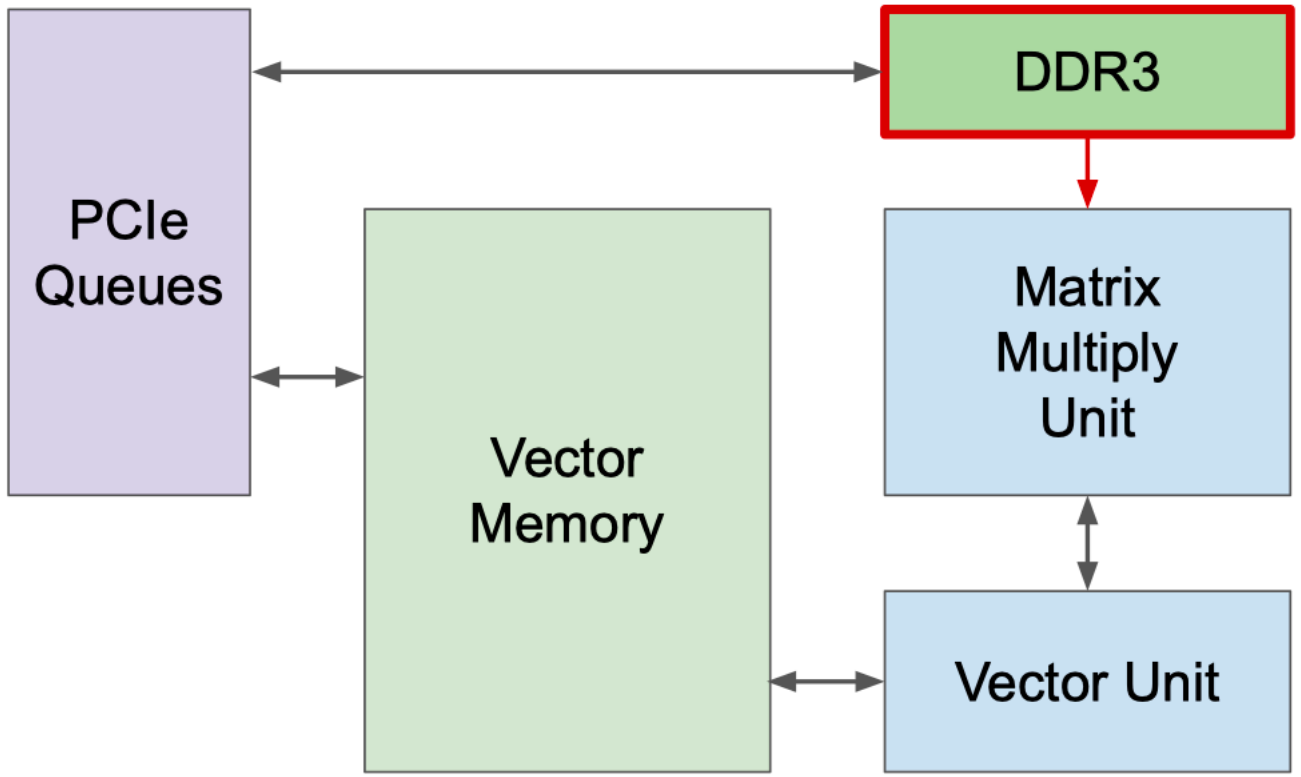
TPUv2 Changes 3

- 将矩阵乘法单元 MXU 直接与向量存储区 VU 连接。
- 如此一来，矩阵乘法单元 MXU 就成为向量单元 VU 的协处理器。
- 这种结构对编译器和编程人员而言更友好。



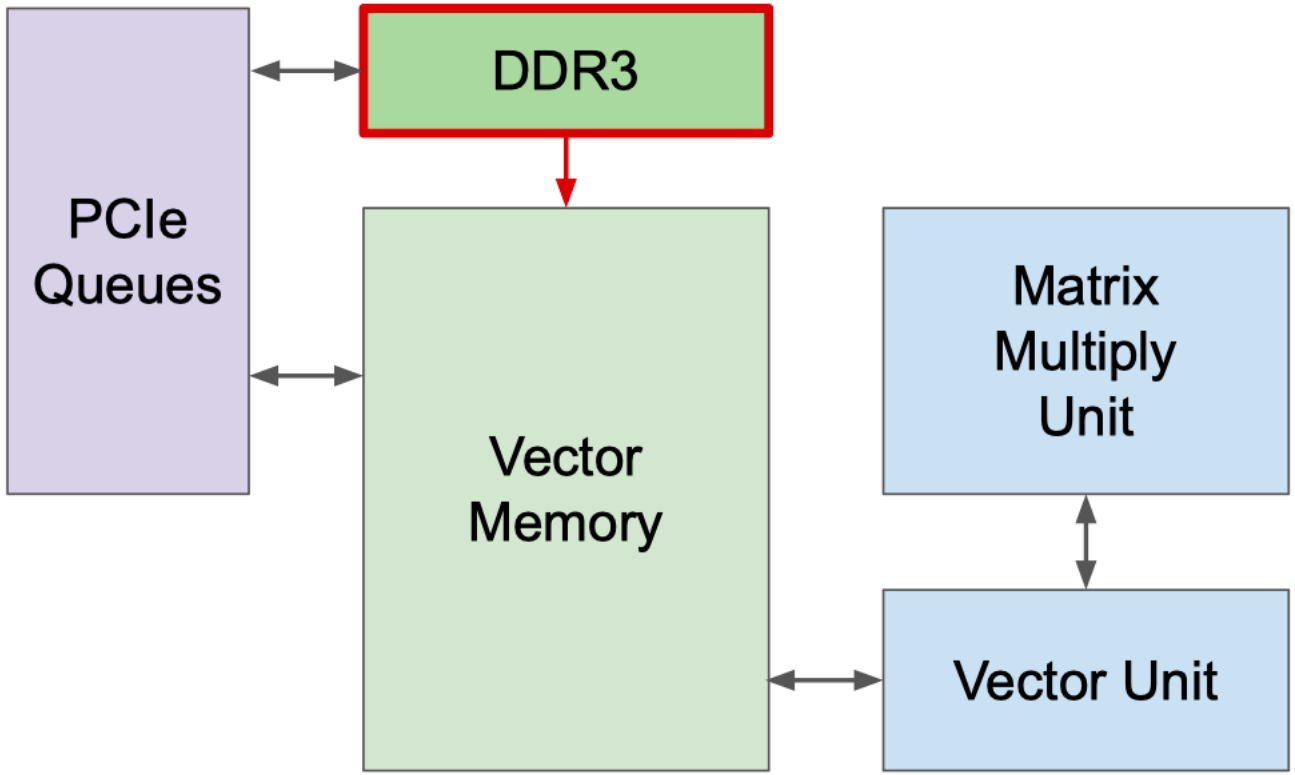
TPUv2 Changes 4

- TPU v1 使用 DDR3 内存，因为它针对的是推理，只需使用已有的权重，不需要生成权重。
- TPU v2 针对训练场景，既要读取权重，也要写入权重，所以在 v2 将原本 DDR3 改为与向量存储区相连，既能向其读取数据，又能向其写入数据。



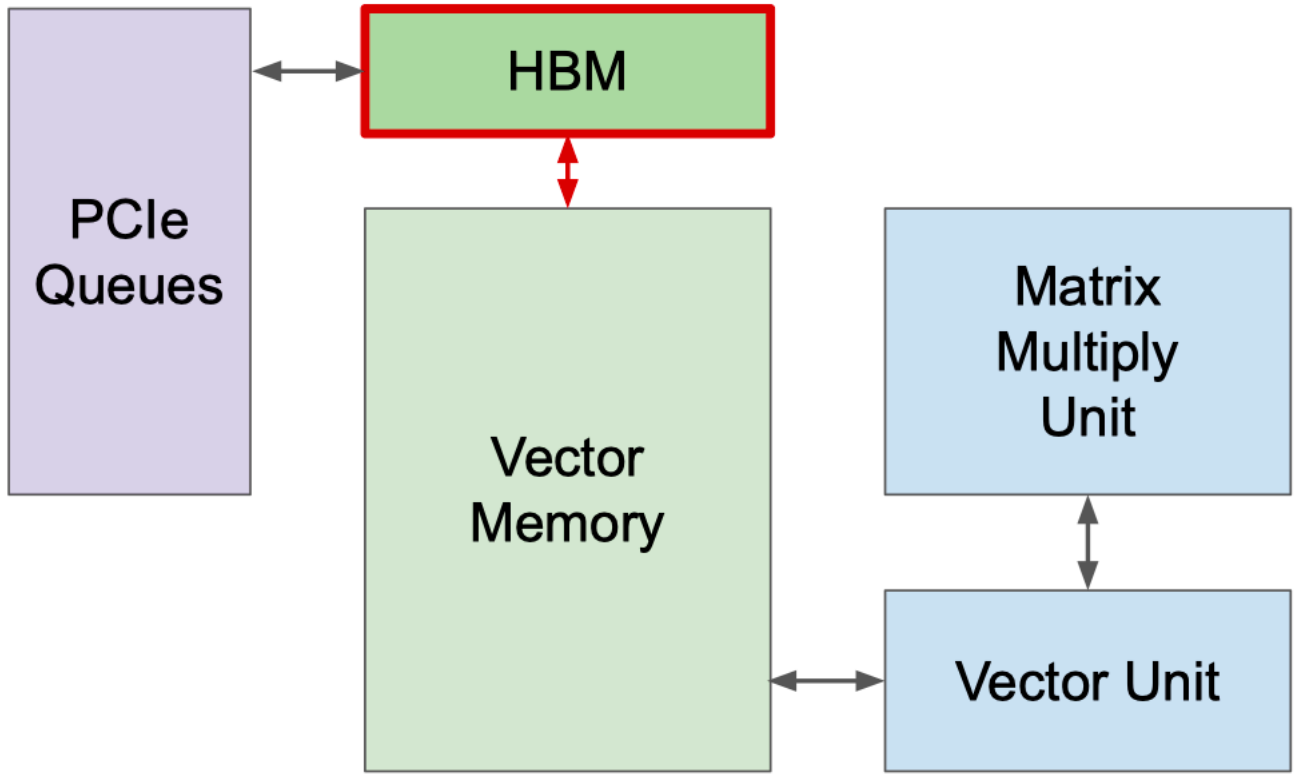
TPUv2 Changes 4

- TPU v1 使用 DDR3 内存，因为它针对的是推理，只需使用已有的权重，不需要生成权重。
- TPU v2 针对训练场景，既要读取权重，也要写入权重，所以在 v2 将原本 DDR3 改为与向量存储区相连，既能向其读取数据，又能向其写入数据。



TPUv2 Changes 5

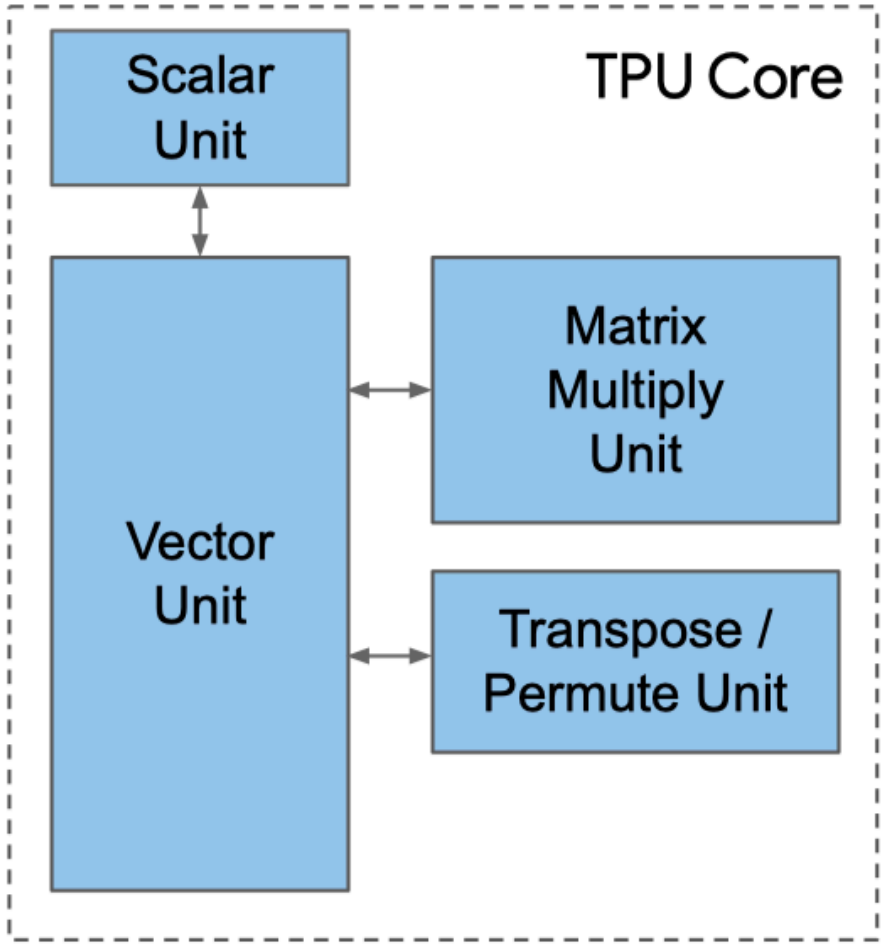
- 将 DDR3 改为 HBM，从 DDR3 读取参数速度太慢，影响性能，而 HBM 读写速度快 20 倍。



2. TPU2 核心

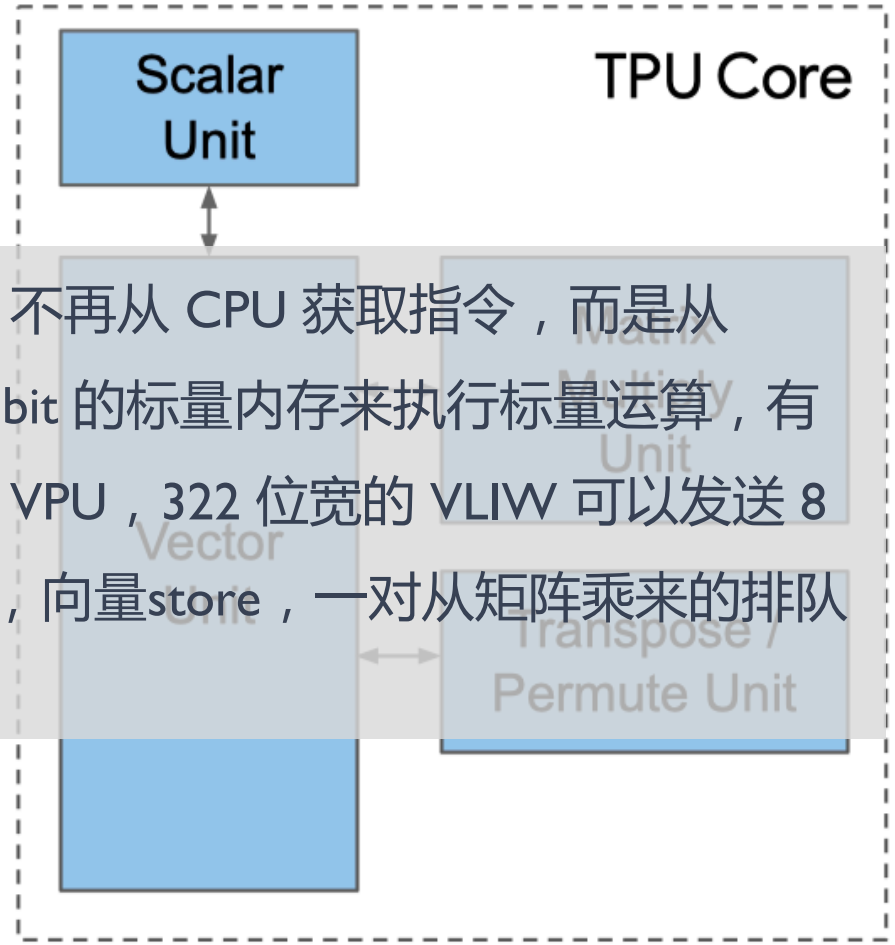
TPU 计算核心：MXU、Scalar Unit

- TPU V2采用了超长指令集架构（ VLIW Architecture ）来提供具体的执行指令（ 322b bundle ）：
  - 2 scalar slots
  - 4 vector slots (2 for load/store)
  - 2 matrix slots (push, pop)
  - 1 misc slot
  - 6 immediates



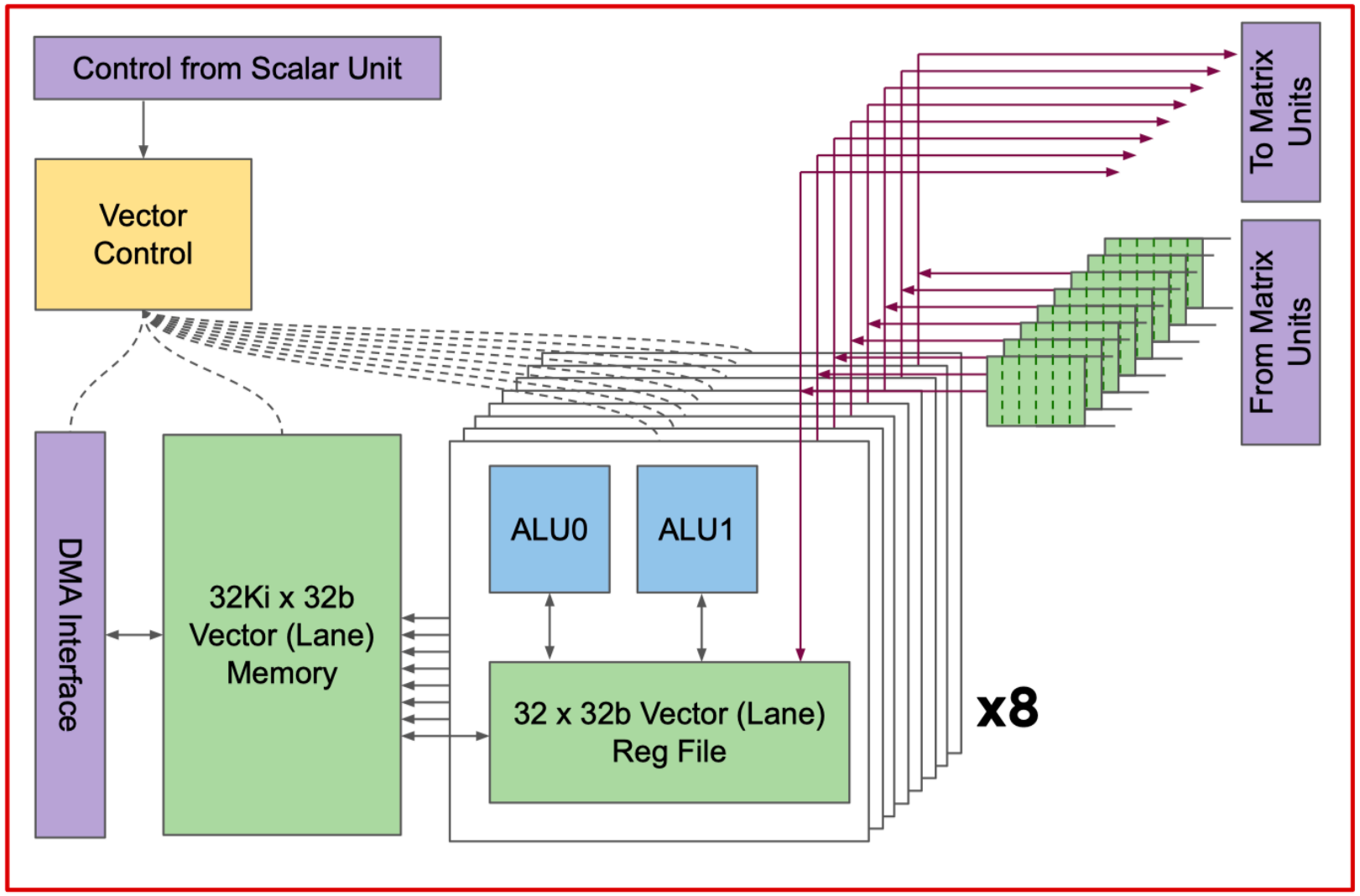
TPU 计算核心：MXU、Scalar Unit

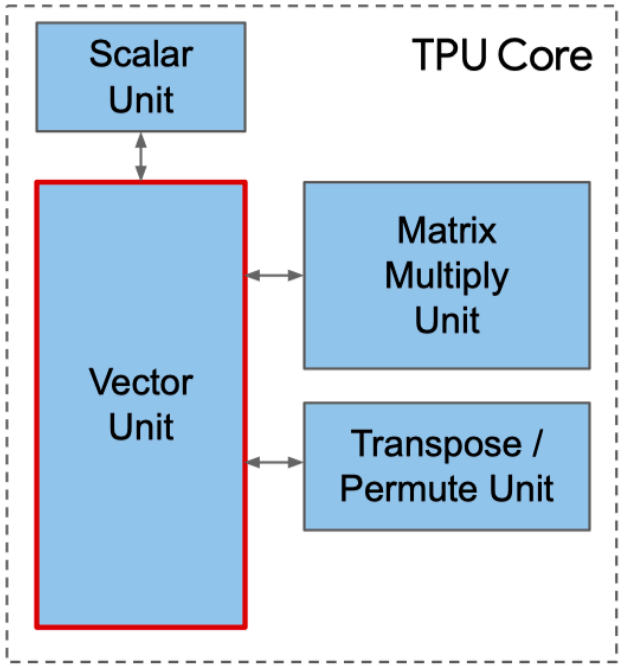
- TPU V2采用了超长指令集架构（ VLIW Architecture ）来提供具体的执行指令（ 3228 Bundle ）：
  - 2 scalar slots
  - 4 vector slots (2 for load/store)
  - 2 matrix slots (push, pop)
  - 1 misc slot
  - 6 immediates



核发射单元 Core Sequencer 。 Core Sequencer 不再从 CPU 获取指令，而是从 Instruction Mem 取出 VLIW 指令，使用 4K 32-bit 的标量内存来执行标量运算，有 32 个 32 位的标量寄存器，而将向量指令送到 VPU，322 位宽的 VLIW 可以发送 8 个操作，2 个标量，2 个向量 ALU，向量 load，向量store，一对从矩阵乘来的排队数据

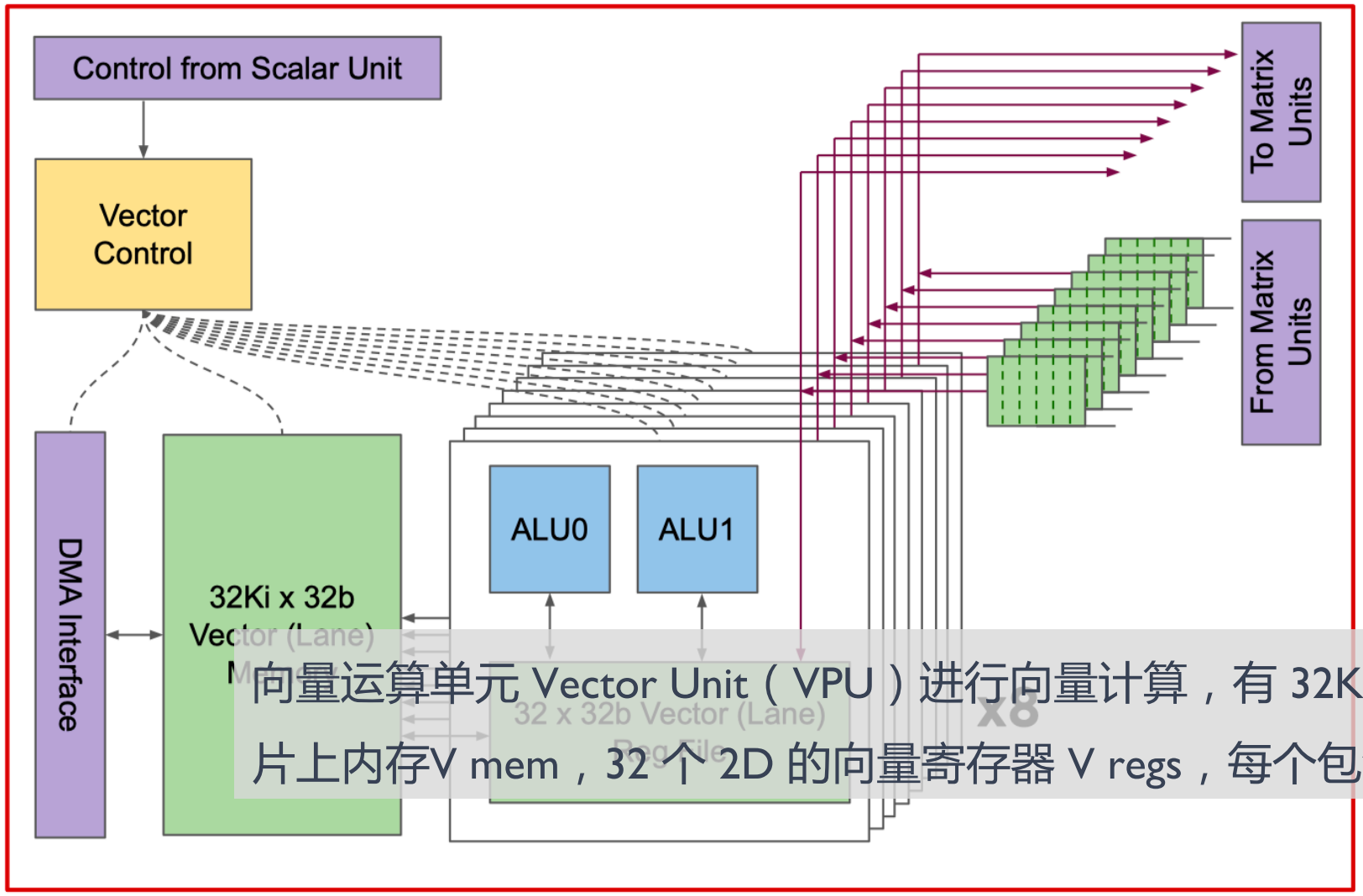
TPU Core: Vector Unit (Lane)

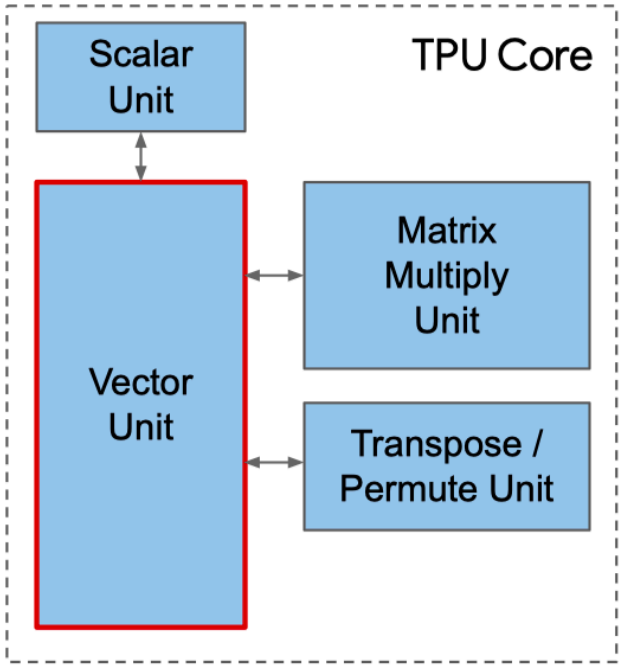




为应对 batch normalization，增加了一个 SIMD 维度给 vector unit；

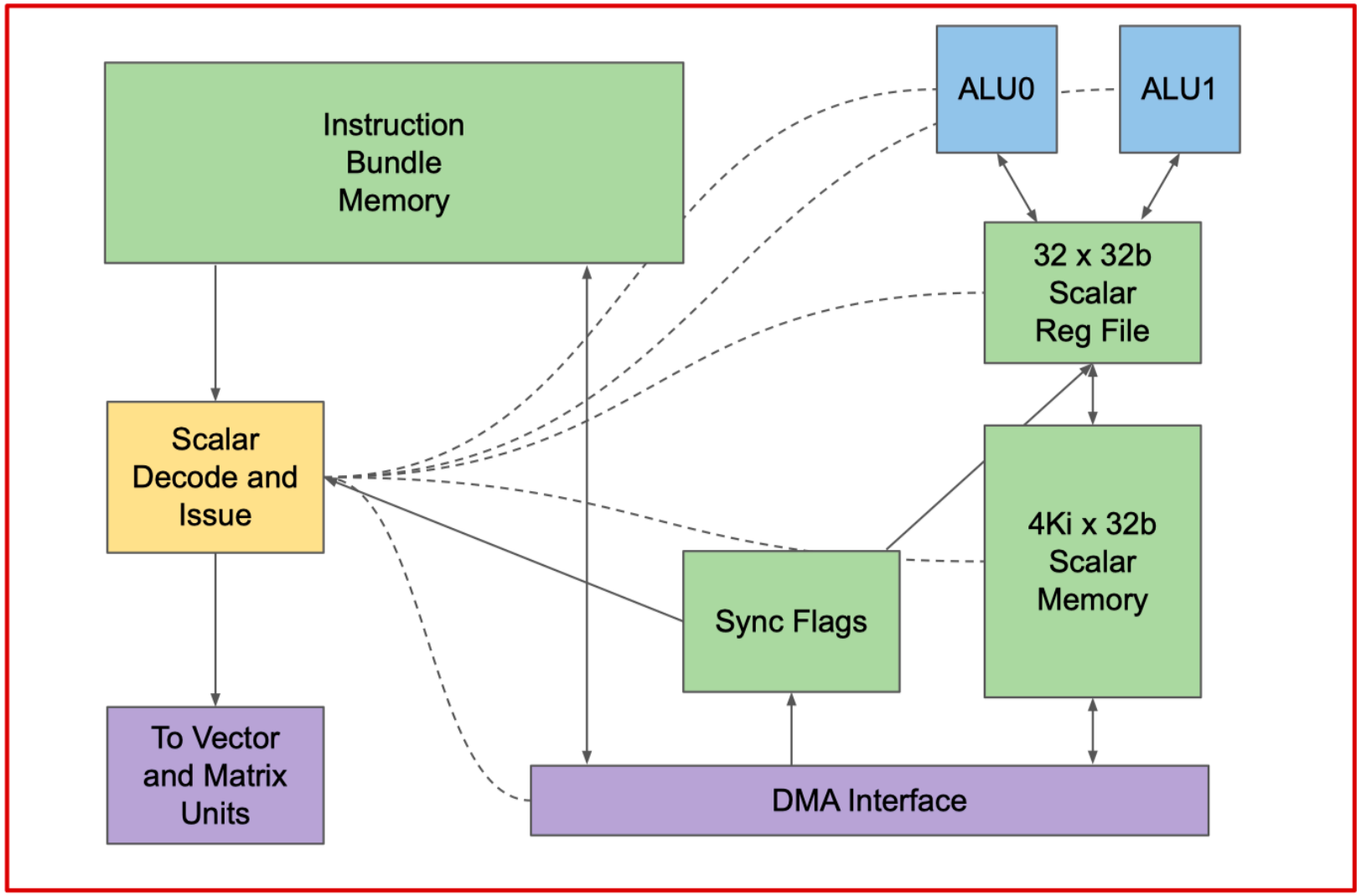
TPU Core: Vector Unit (Lane)

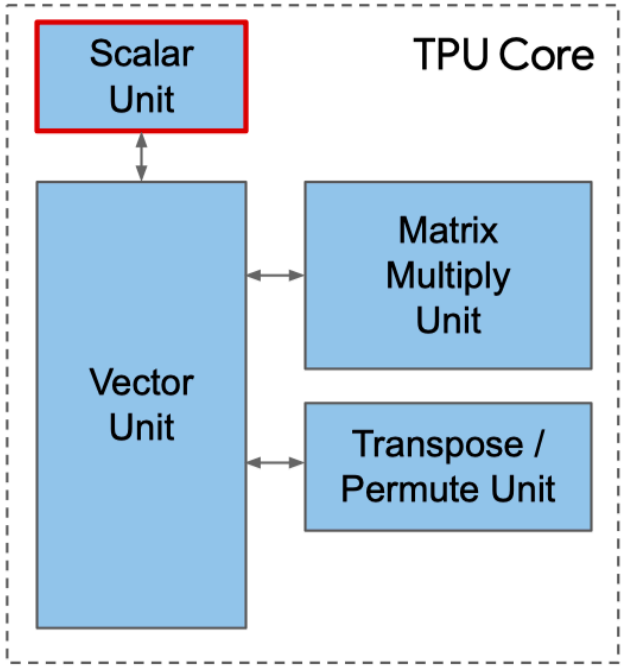




向量运算单元 Vector Unit（VPU）进行向量计算，有 32K 128x32 位的16MiB 大小的片上内存V mem，32 个 2D 的向量寄存器 V regs，每个包含128x8 32位元素（4KiB）

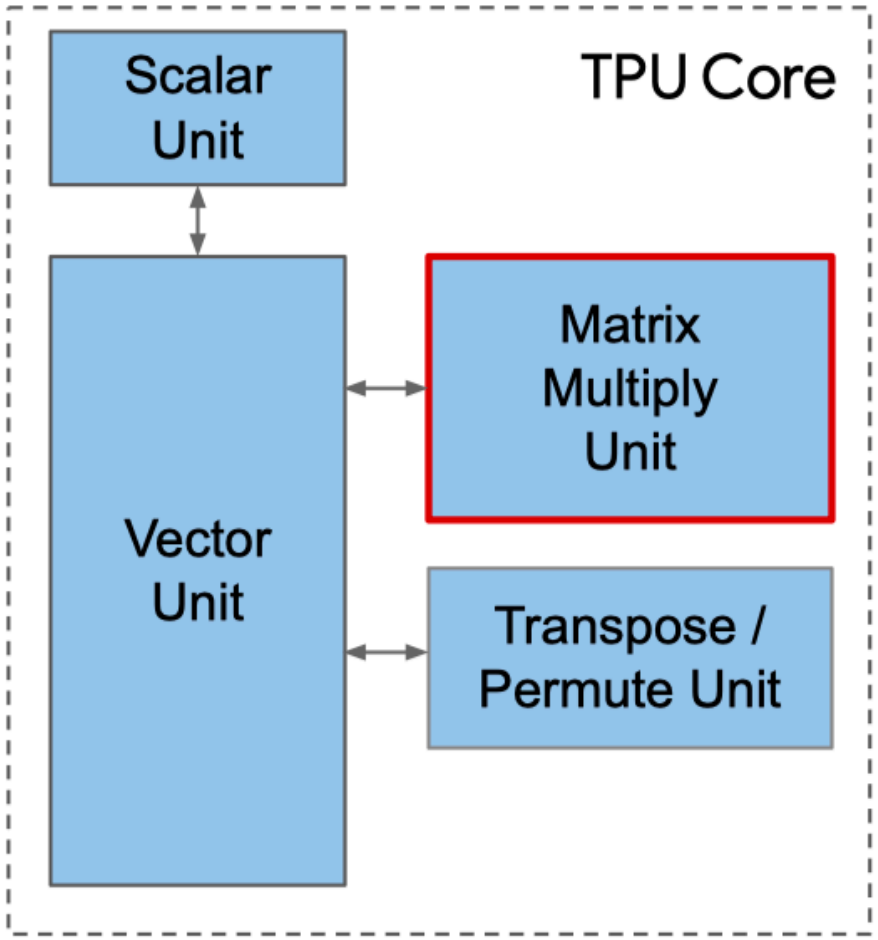
TPU Core: Scalar Unit





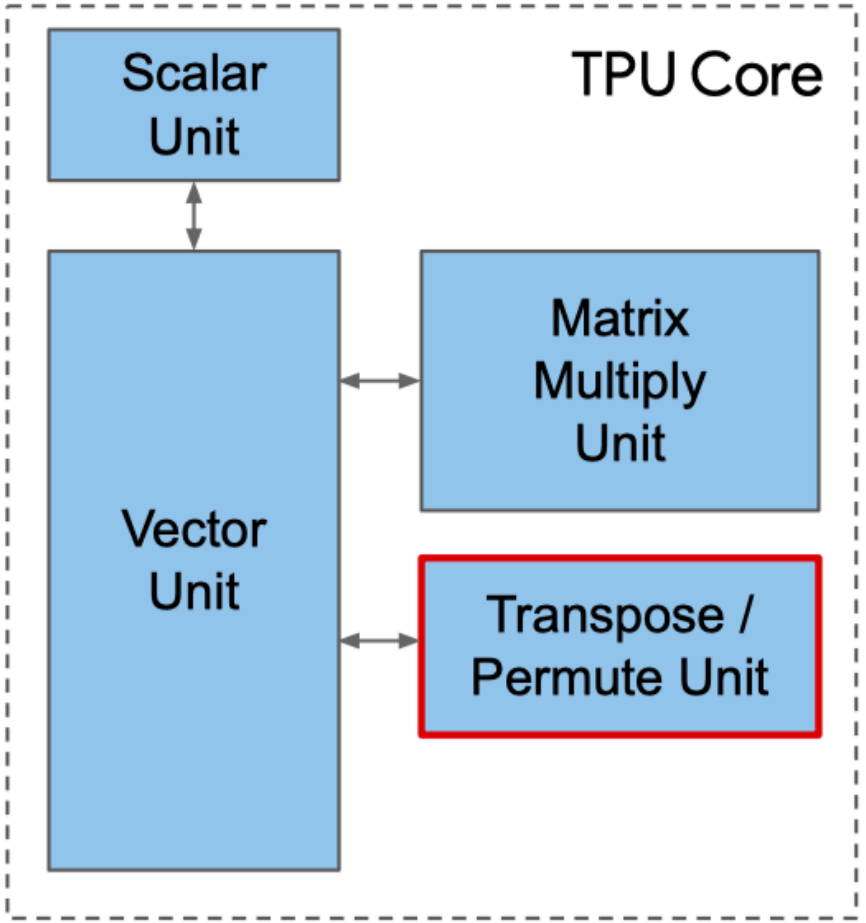
TPU Core: Matrix Multiply Unit

- 矩阵运算单元 MXU，输入 16 位的 FP 得到 32 位 FP 结果，一个芯片有多个 MXU脉动阵列，每个大小为 128 x 128：
  - Streaming LHS and results
  - Stationary RHS (w/optional transpose)
- 计算数值支持 BF16 计算和 FP32存储：
  - Bfloat16 multiply {s, e, m} = {1, 8, 7}
  - Float32 accumulation



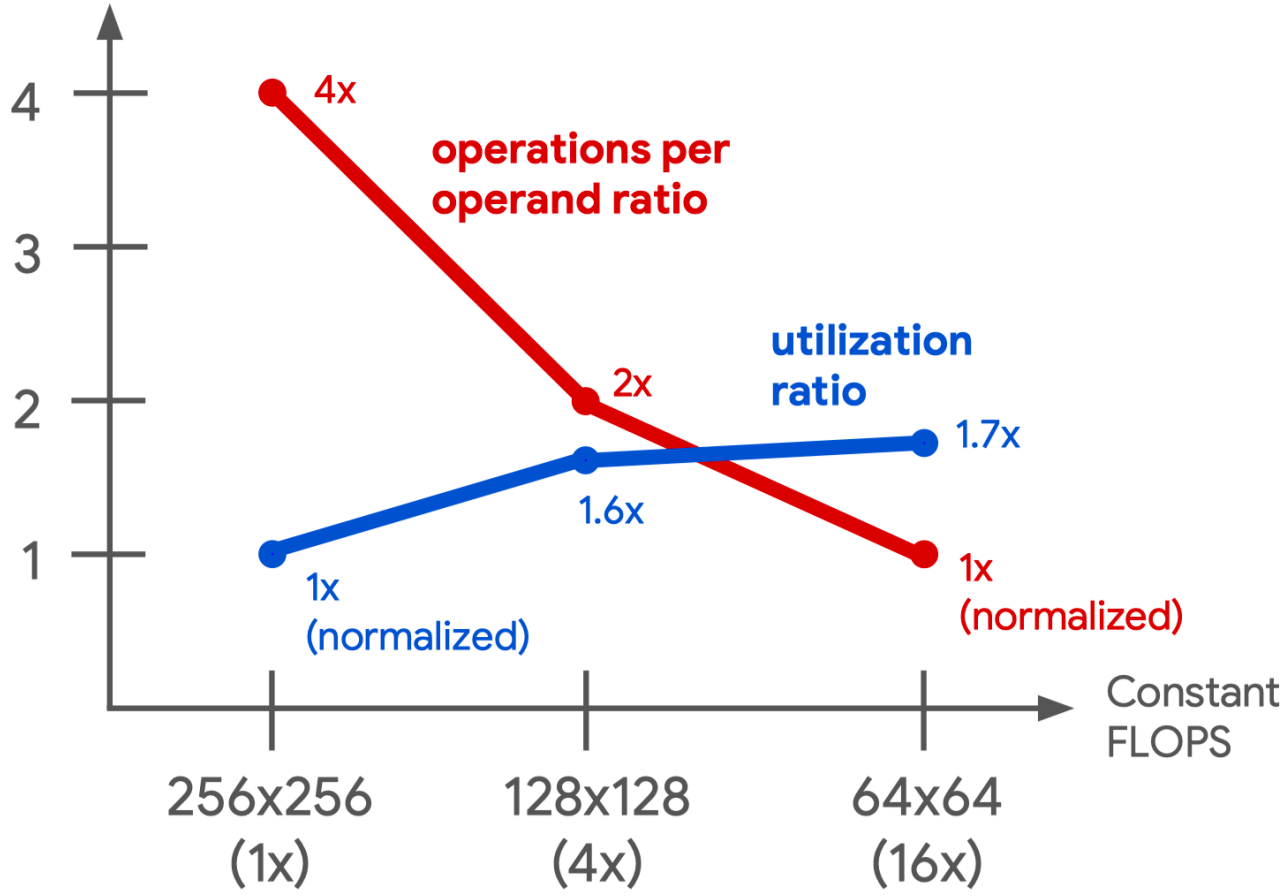
TPU Core: Transpose, Reduction, Permute Unit

- 矩阵特殊计算操作，允许矩阵数据进行重新排布，提升编程易用性。
  用于做 128x128 矩阵进行操作：
  - Transpose 矩阵转置
  - Reduction矩阵规约
  - Permutation 置换矩阵



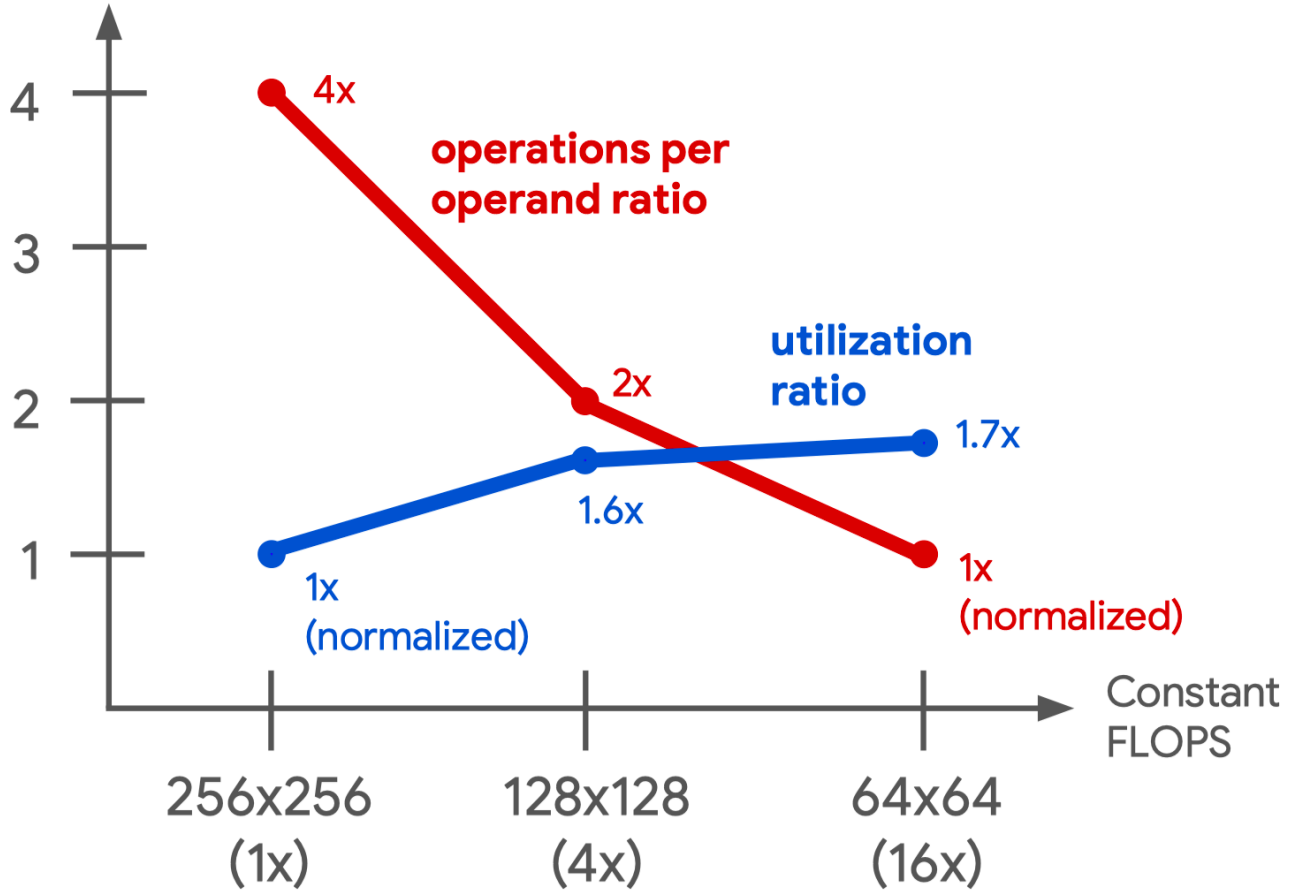
思考

- Why 128x128?



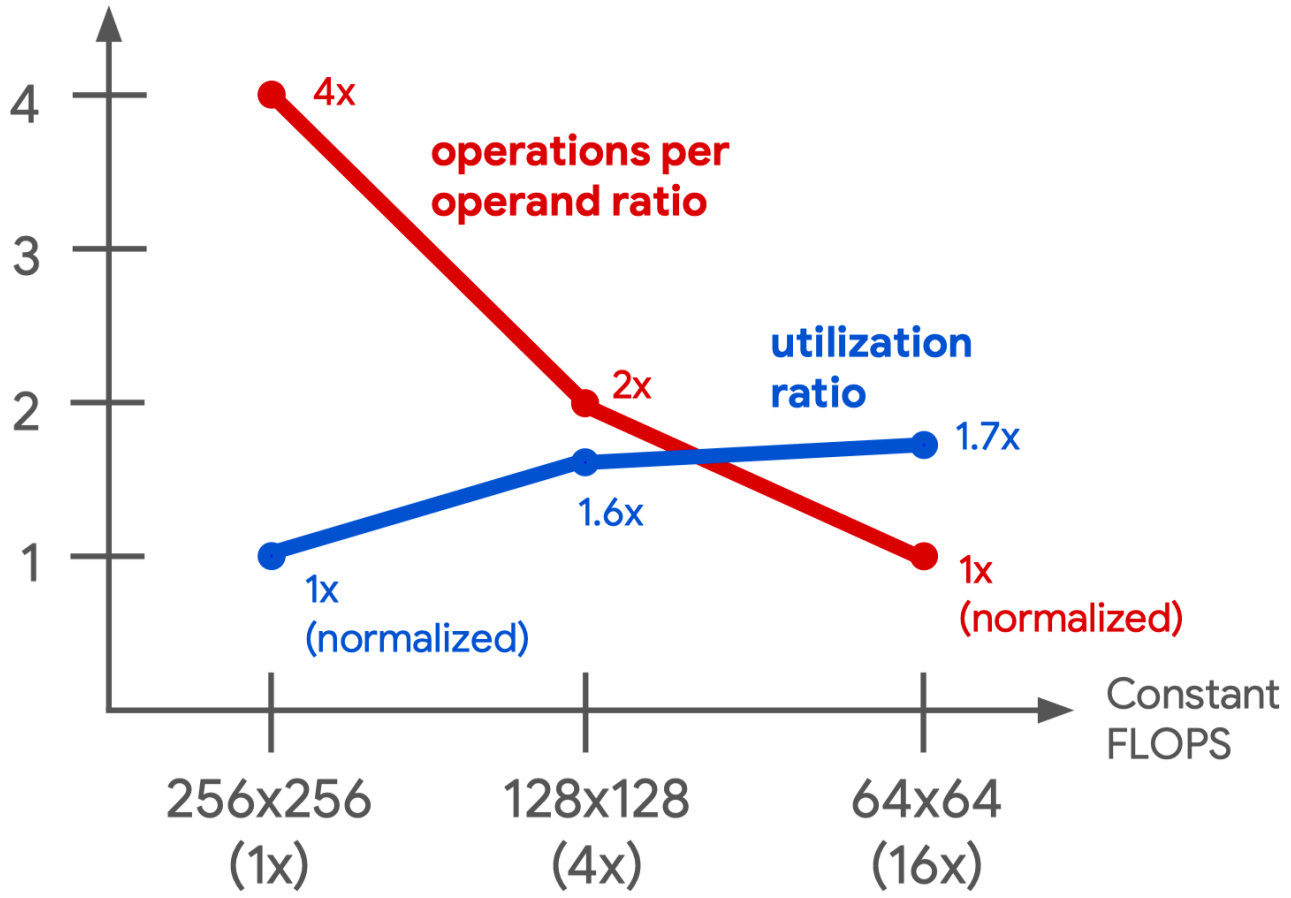
MXUs Core 的选择

- **核心位宽选择：**TPU v2 两个核，单个核 MXU 相比 TPU v1 小了四分之一，有些卷积计算 256x256 单用一个大核，会导致 MXU 利用率不高；使用 16 个 64x64 MXUs 需要更多芯片电路面积，因此选择两个 128x128。



MXUs Core 的选择

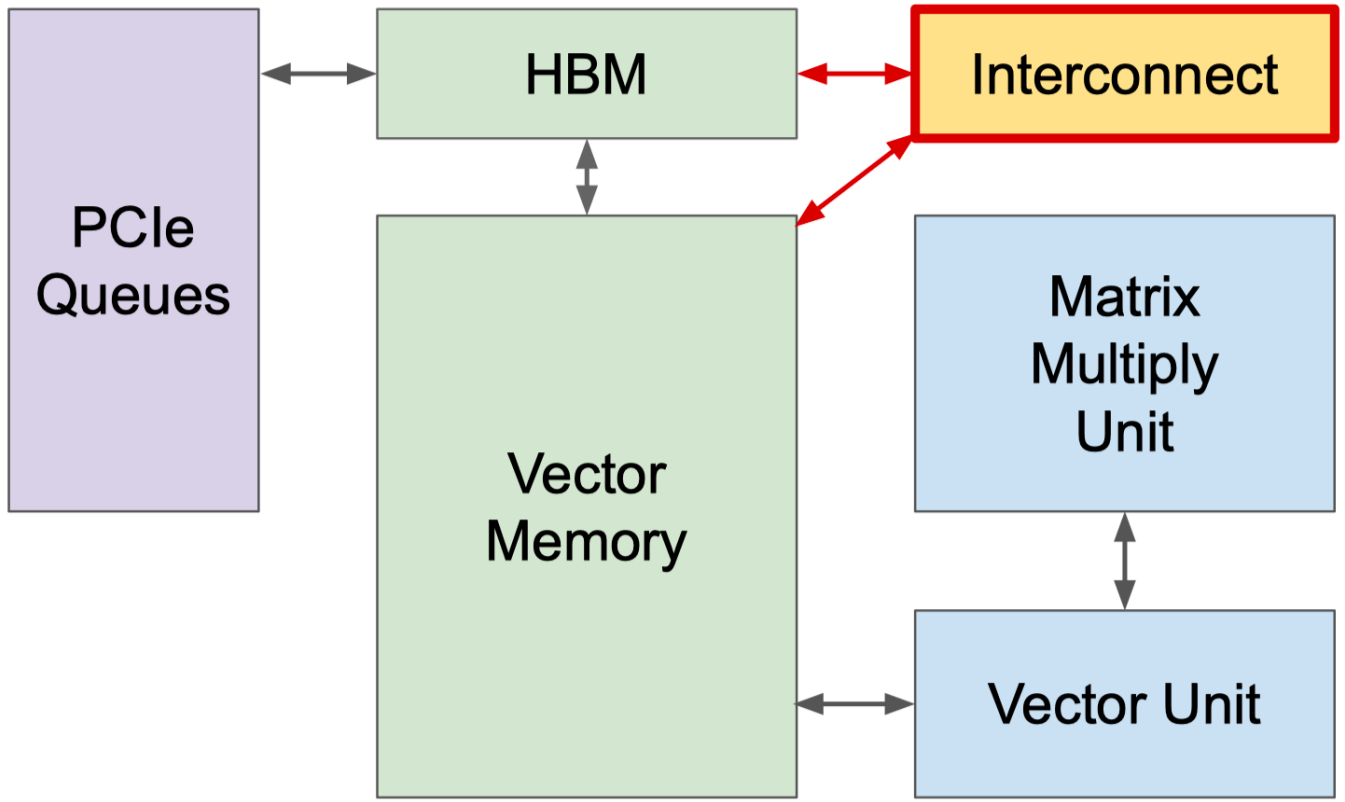
- MXU 输入和输出结果所需的带宽与其周长成正比，但它提供的计算量却与面积成正比，与周长的平方成正比。
- 因此越大的 MXU 其受限于带宽瓶颈的问题越明显。另一个理由是，单个大核下，脉动阵列过大会产生较大延迟，且芯片上核数过多也会增加编译程序的复杂度。



3. 内存与互联

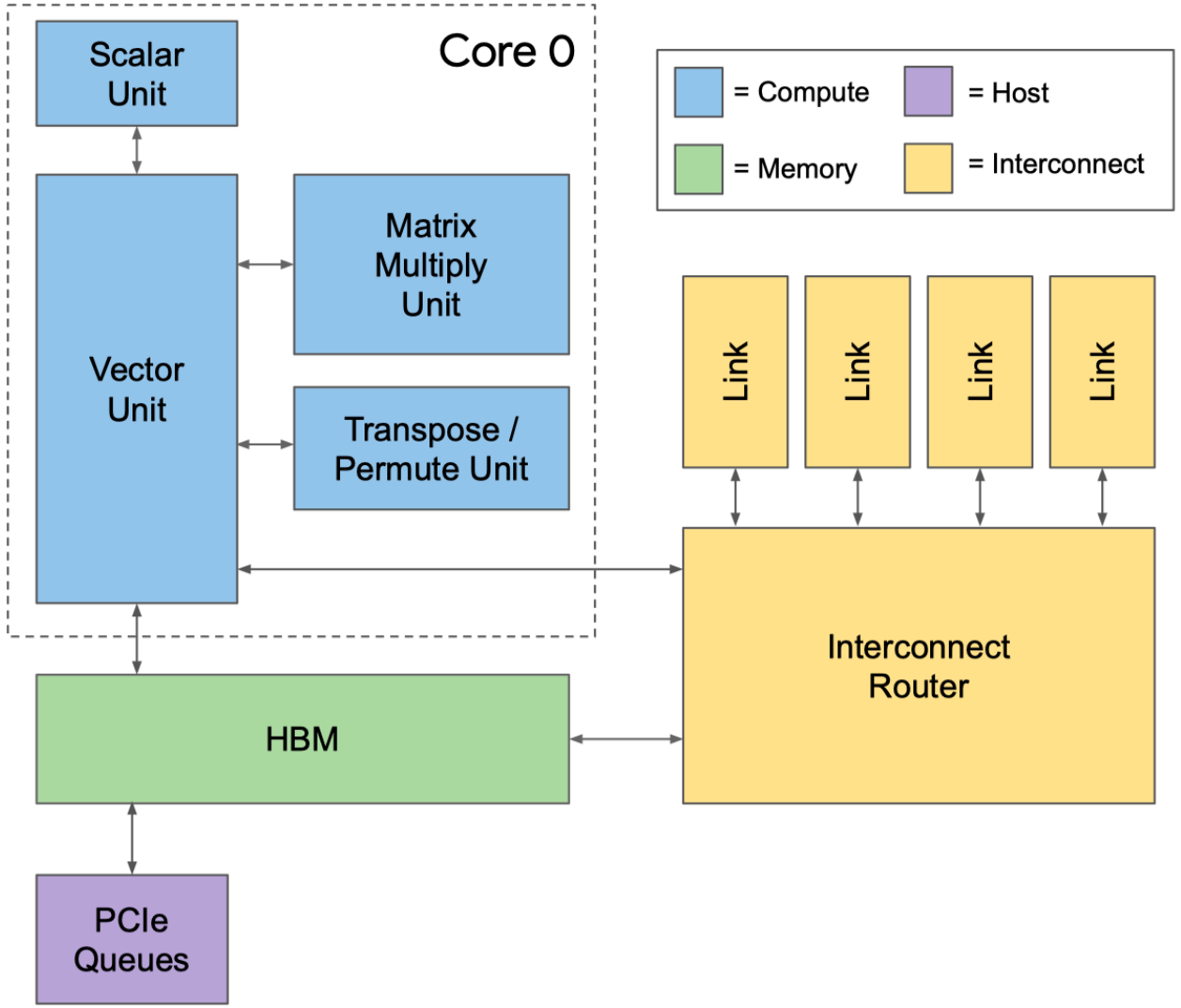
TPUv2 新的互联方式

- 在 HBM 和向量存储区之间增加互连（Interconnect），用于TPU之间的连接，组成 Pod 超级计算机。



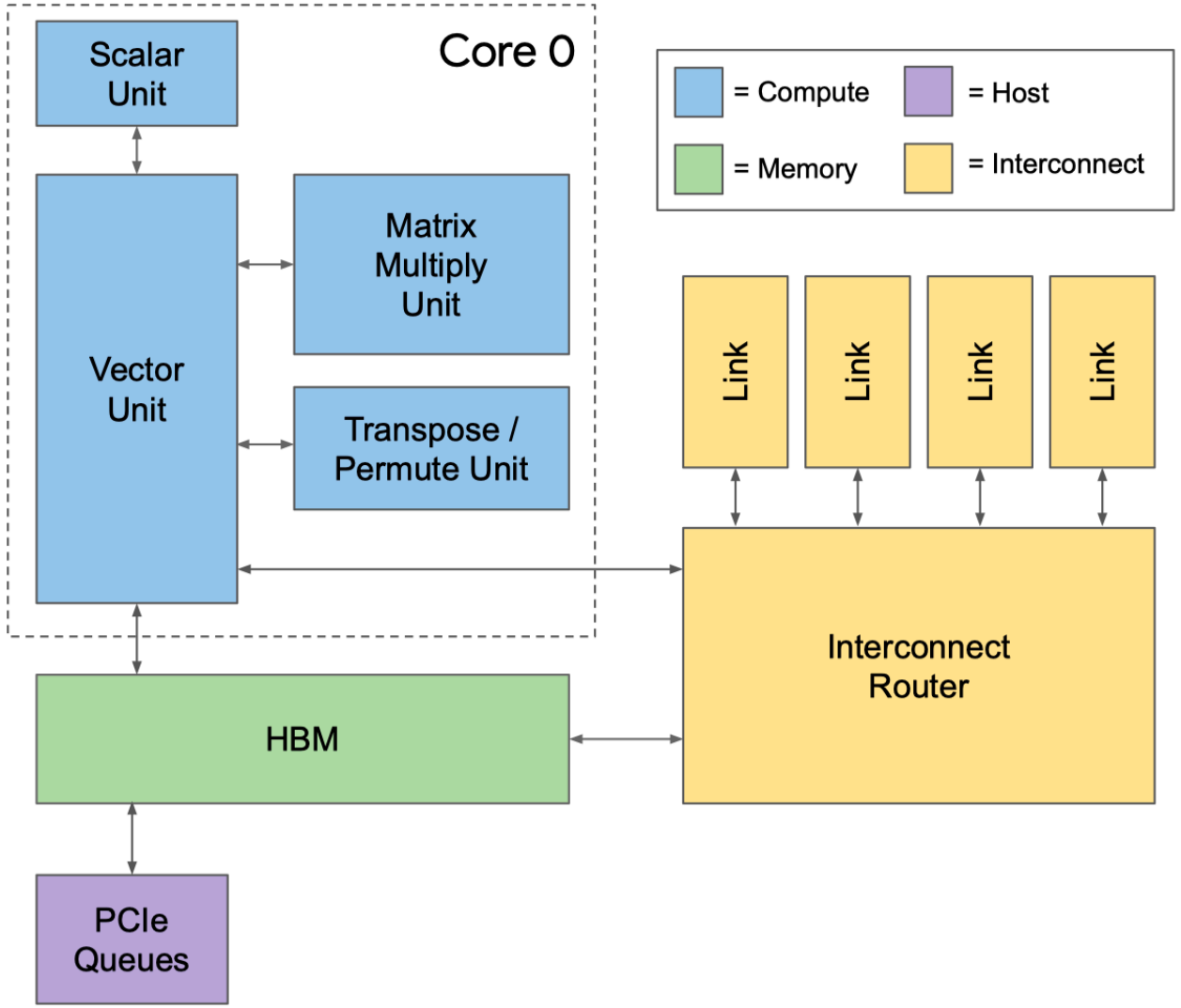
TPUv2 新的互联方式

- 高带宽内存 High Bandwidth Memory, HBM：TPU v1 中的大多数神经网络都受到内存限制，故使用 HBM。
- 使用一个中间衬底，通过 32 条 128 bit 总线将 TPUv2 芯片连接到 4 个短 DRAM 芯片堆栈，从而提供了 20 倍于 TPUv1 的带宽。

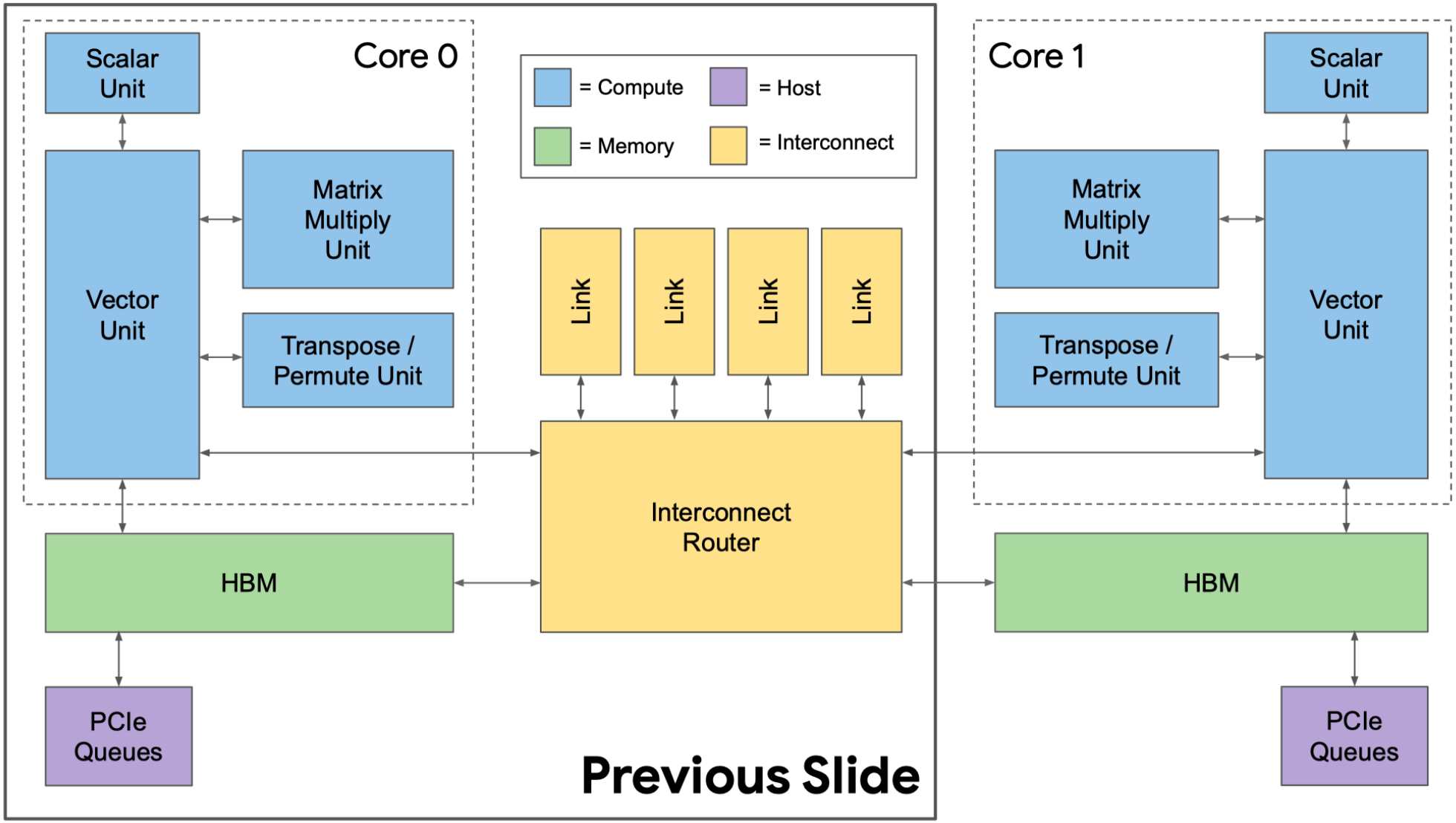


TPUv2 新的互联方式

- 核间互联 Interconnect Router：为了实现 2D 环面连接，以组成 Pod 超级计算机，芯片有四个自定义的核间互连（ICI）链路，每个链路运行在 TPUv2 中，每方向 496Gbits/s。
-



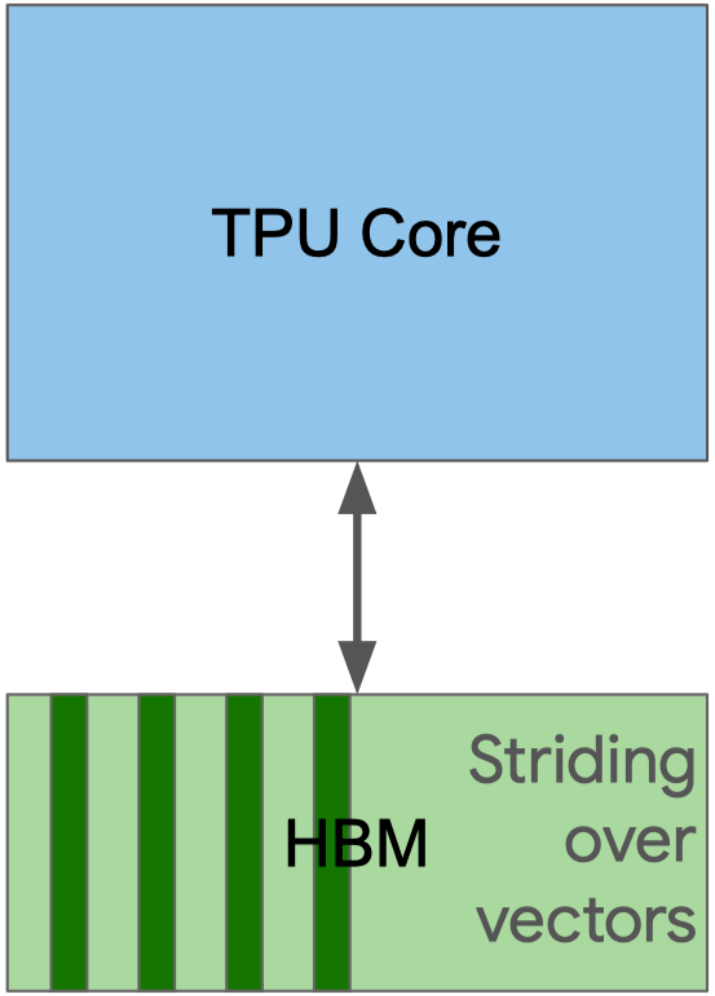
TPUv2 新的互联方式



Memory System

- Loads and stores against SRAM scratchpads
- Provides predictable scheduling within the core
- Can stall on sync flags

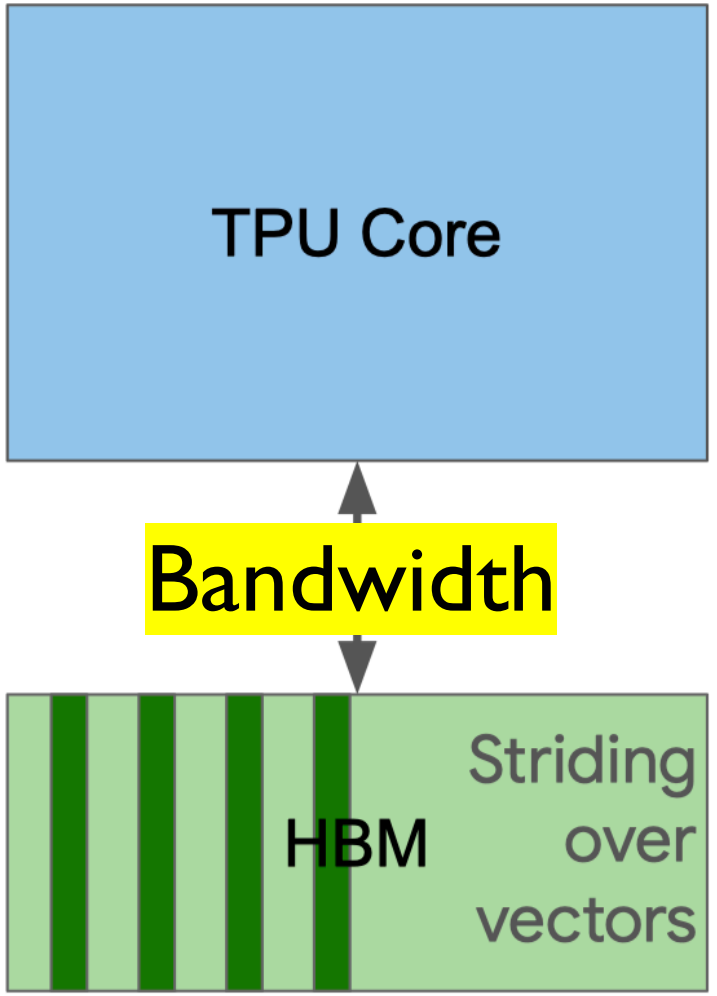
- Accessible through asynchronous DMAs
- Indicate completion in sync flags



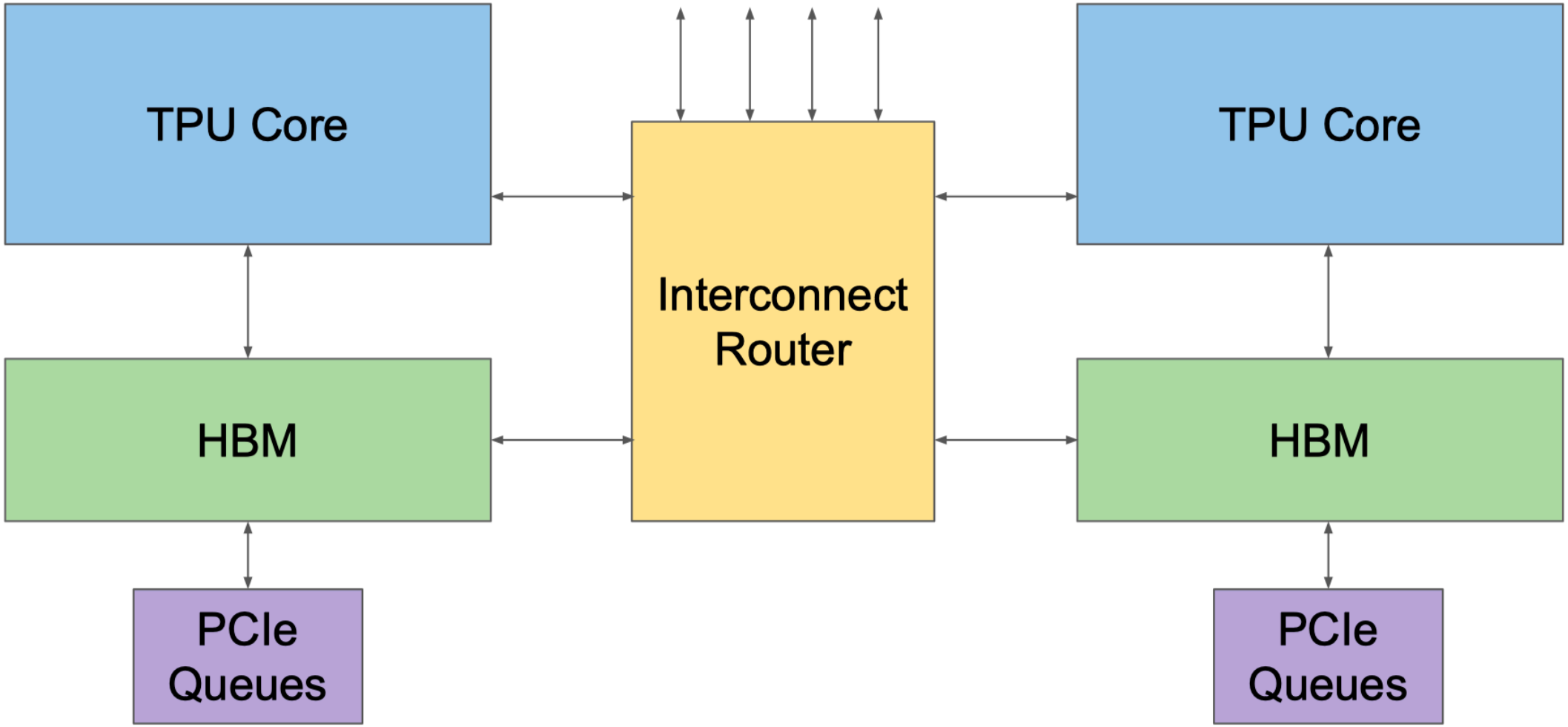
Memory System

- Loads and stores against SRAM scratchpads
- Provides predictable scheduling within the core
- Can stall on sync flags

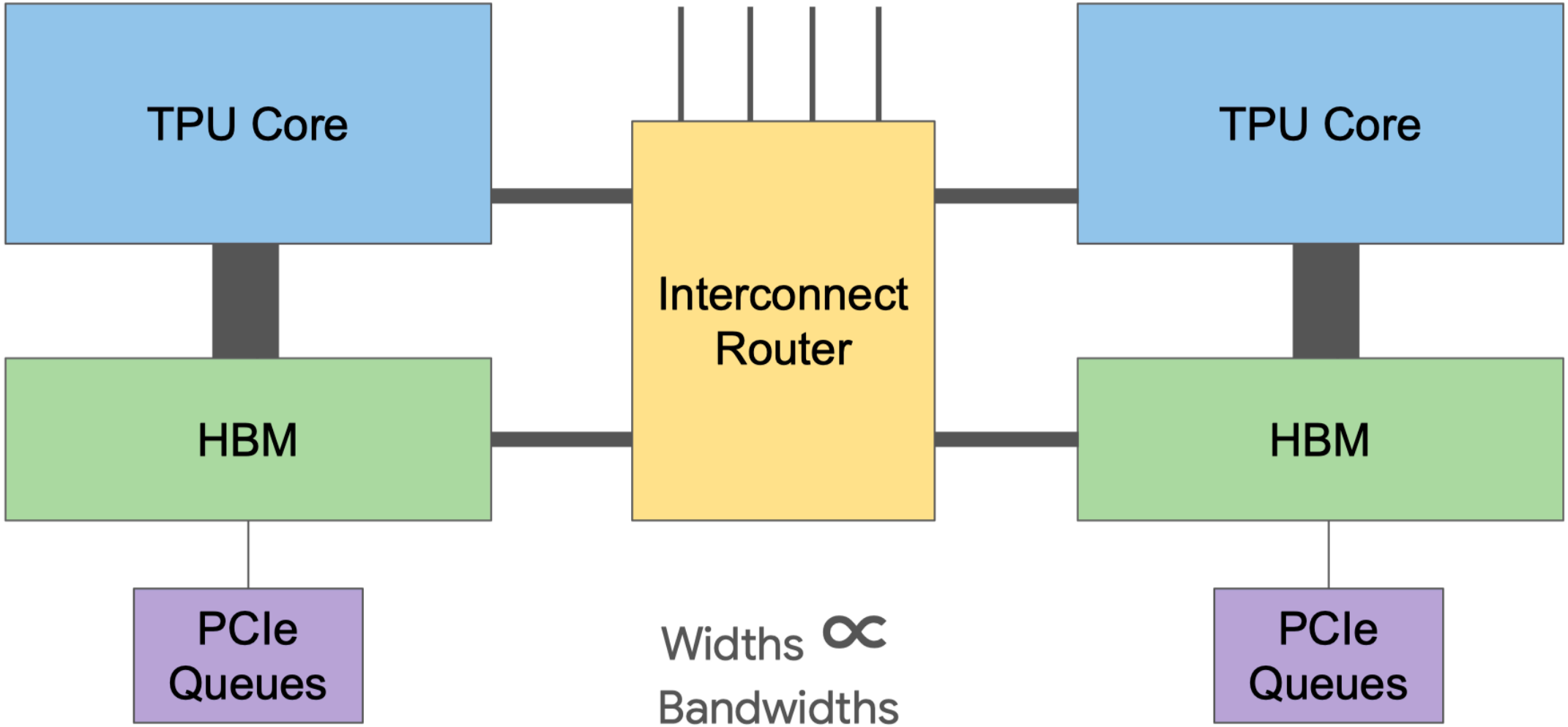
- Accessible through asynchronous DMAs
- Indicate completion in sync flags



Memory System

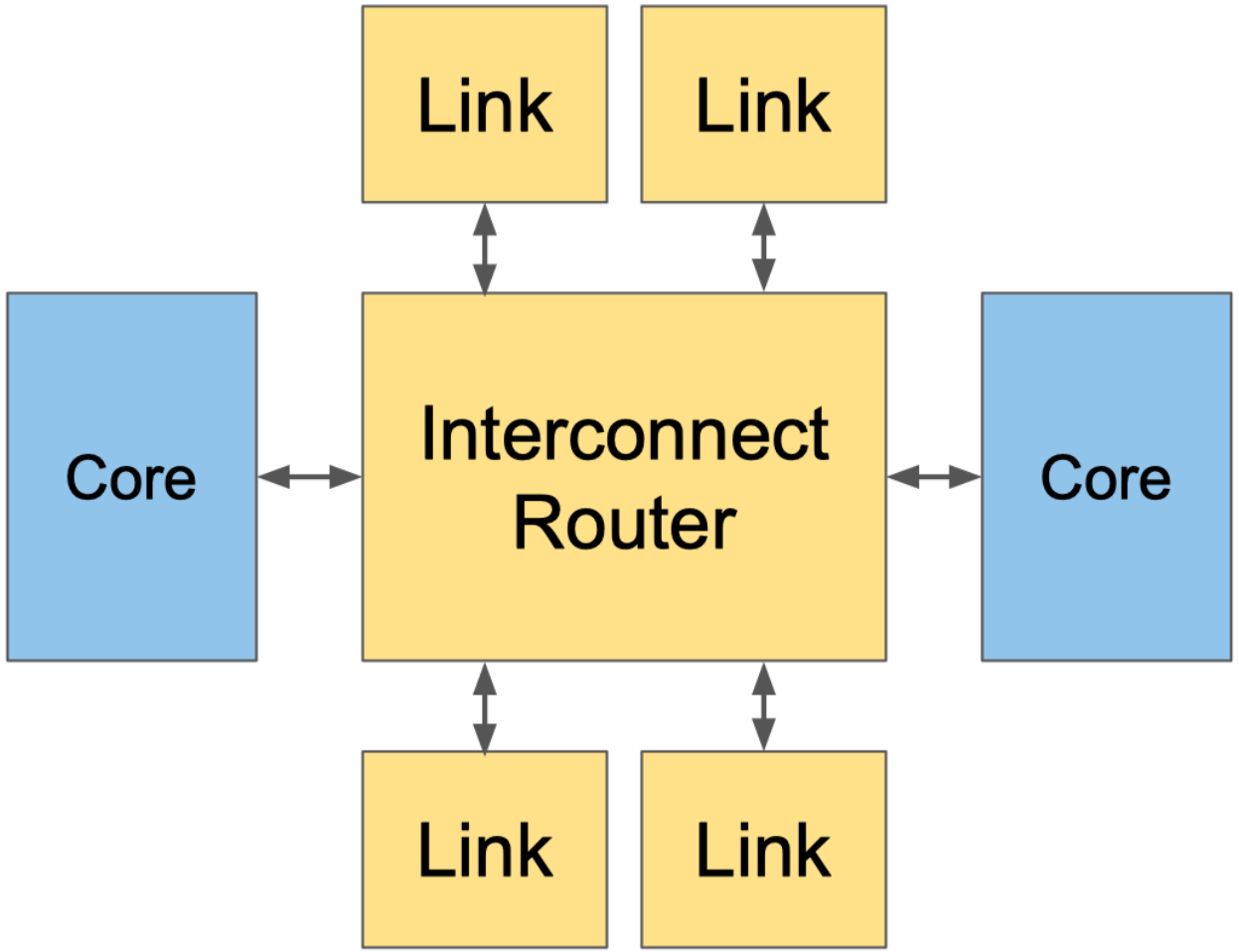


Memory System



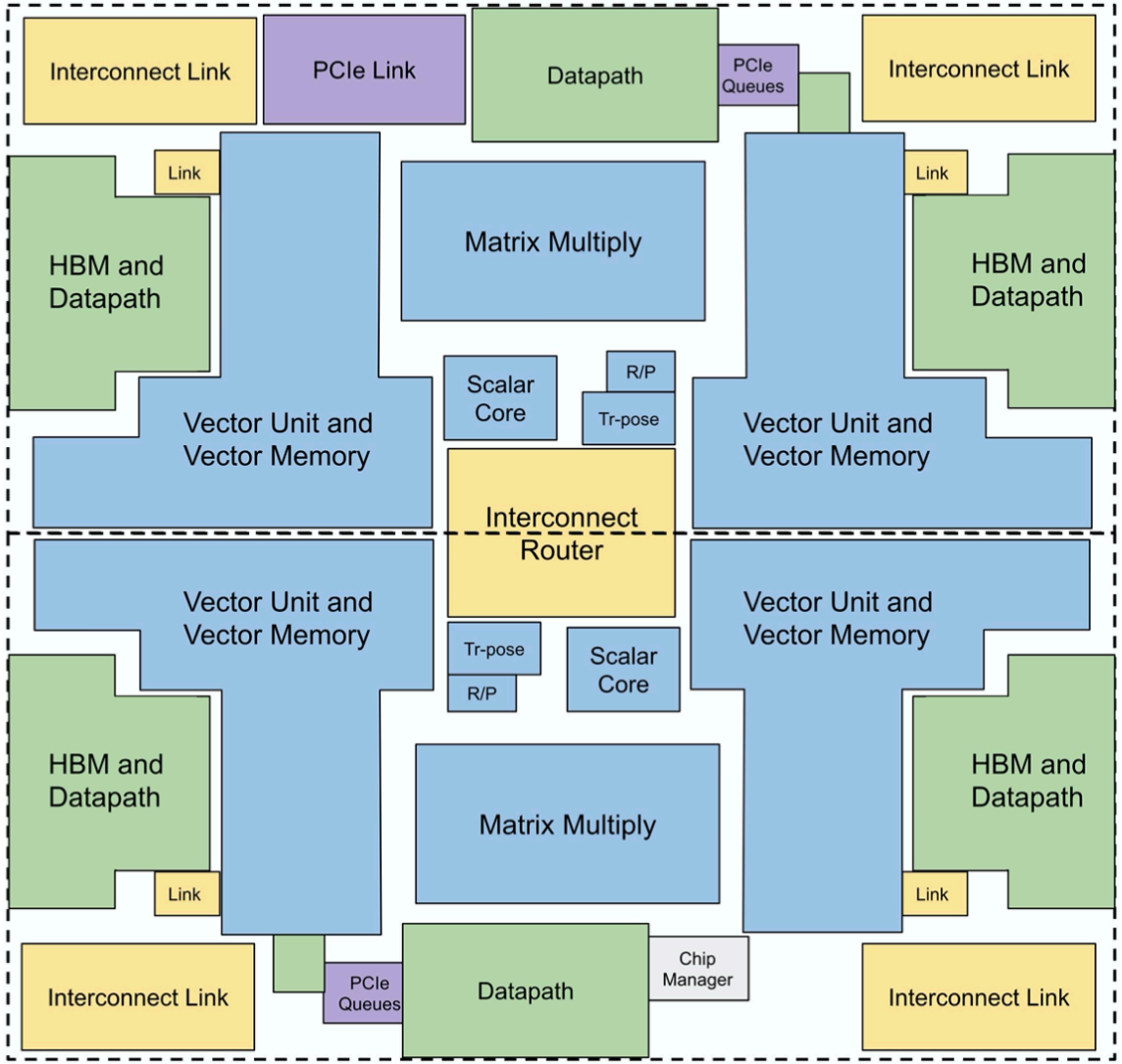
Interconnect

- On-die router with 4 links
- 500 Gbps per link
- Assembled into 2D torus Software view:
  - Uses DMAs just like HBM
  - Restricted to push DMAs
  - Simply target another chip id

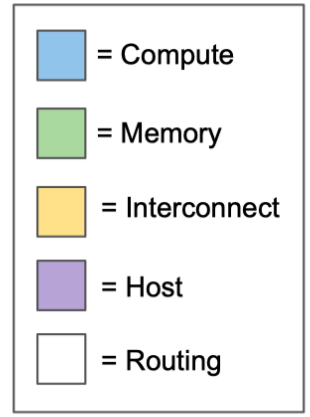


4. 总结与思考

TPU2 芯片布局图：专用电路和大量缓存，提供2个计算核心



Floorplan



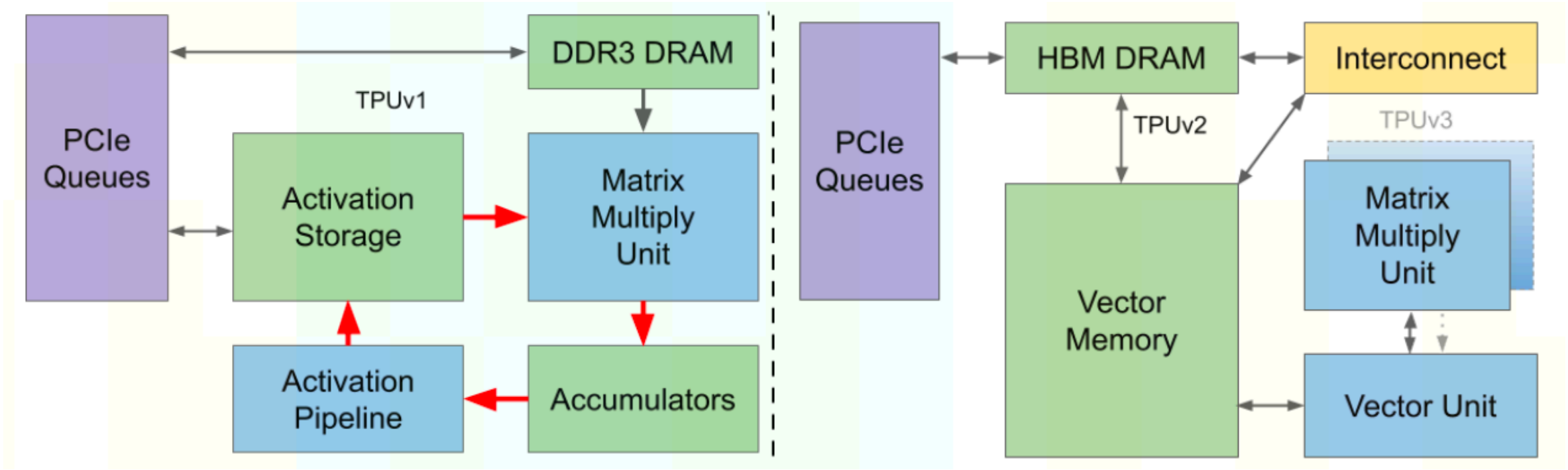
TPU v2特点总结

1. TPU v2 每块芯片中提供两个 Tensor Core

2. 将固定激活函数（ Activation Pipeline ）改为可编程性更高的向量单元（ Vector Unit ）
3. 使用向量存储器（ Vector Memory ）代替 Accumulator 和 Activation Storage 中的双缓存
4. MXU 作为向量单元的协处理器直接与向量单元连接，增加其可编程性
5. 增加 Scalar Unit, Transpose/Permute Unit 等特殊计算单元

6. 使用HBM代替DDR3，并改为与向量存储区相连，提供更高的带宽和读写速度
7. HBM和向量存储区之间增加互连模块（Interconnect），使TPU间提升互联带宽

TPU v1 vs TPU v2



Thank you

把AI系统带入每个开发者、每个家庭、每个组织，构建万物互联的智能世界

Bring AI System to every person, home and organization for a fully connected, intelligent world.

ZOMI

Course chenzomi12.github.io

GitHub github.com/chenzomi12/DeepLearningSystem